# PCS 91.S7
# english

**One does not refer in the manual explicitly to the devices of the PCS plus/win series, the description applies to all devices. With differentiations between the equipment series the following allocations apply:**

| | | | |
|---|---|---|---|
| **PCS topline** | **=** | **micro/mini:** | **PCS 009, PCS 090, PCS 095, PCS 095.1, PCS 095.2** |
| | | **midi:** | **PCS 900, PCS 950, PCS 950c, PCS 950q, PCS 950qc,** |
| | | **maxi:** | **PCS 9000/9100** |
| **PCS plus** | **=** | **micro/mini:** | **PCS 009 plus, PCS 090 plus, PCS 095 plus** |
| | | **midi:** | **PCS 950 plus, PCS 950c plus, PCS 950q plus,** |
| | | | **PCS 950qc plus** |
| **PCS win** | **=** | **micro/mini:** | **PCS 009 win, PCS 090 win, PCS 095 win** |
| | | **midi:** | **PCS 950 win, PCS 950c win, PCS 950q win,** |
| | | | **PCS 950qc win** |

Reg 10132/0399

CiS-Nr.: 360.110.0480
Version 2/03.03

Systeme Lauer GmbH & CoKG
Postfach 1465
D-72604 Nürtingen

Operator reference manual: PCS 91.S7
Version: 07. February 2003
Person responsible: Zoch



- We reserve the right make changes to the reference manual without prior notice.
- We can not guarantee the accuracy of the programs and data stored on the diskette and the fault-free state of this information.
- Since diskette represent manipulatable data media, we can only guarantee the physical completeness. The responsibility is limited to a replacement.
- At any time, we welcome suggestions for improvements and remarks on errors.
- The agreement also applies to the special appendices to this reference manual.

# Notes for the user

Please read the manual before beginning and keep the manual for later use.

Target group

The manual has been conceived and written for users who are experienced in the use of PCs and automation technology.

Typographical conventions

| | |
|---|---|
| [KEY] | Keys that are to be pressed by the user are given in square brackets, e.g [CTRL] or [DEL] |
| `Courier` | On-screen messages are given in the Courier font, e.g. C:\> |
| **`Courier bold`** | Keyboard input to be made by the user are given in Courier bold, e.g. C:\>DIR |
| *Italics* | Names of buttons to be pressed, menus or other on-screen elements and product names are given in italics. |

Pictograms

The manual uses the following pictograms to highlight certain text passages:

Danger!
Possibly dangerous situation. Injury to persons can be the result.

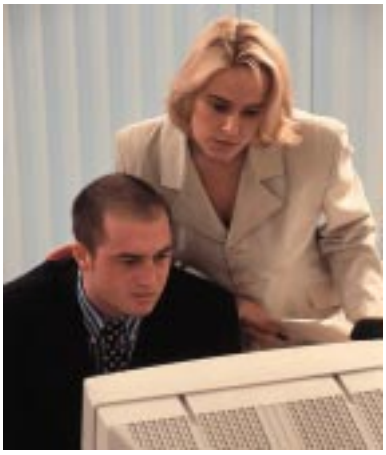
Attention!
Possibly dangerous situation. Property damages can be the result.

Tips and supplementary notes

# Table of Contents

# Quality and support

In our company, quality comes first. From the electronics component up to the finished device, the quality assurance test competently and comprehensively.
National an internation test standards (ISO, TÜV, Germanischer Lloyd) are the basis.
Within 48 hours, every device passes a 100% check and continuous test under worst case conditions at changing temperatures (0...50°C) and test voltages.
A guarantee for maximum quality.



Our products not only feature a maximum economic efficiency and reliability but also a comprehensive complete service.
You not only receive demo devices but we rather make specialists available who support you in person with your first application.
Qualified user consultation by competent sales engineers is obvious for us.
Our support is for you for the side with advice and deed every day.

We set up training programs and technical training for you in our modern training center or alternatively also in your house.
Request the curent training catalog.

From the consultation up to the user support, from the hotline up to the service, from the reference manual up to the training an all covering and individual service for the entire product line is waiting for you.

Whenever you need us, we are there for you:
dynamically, creatively and enormously efficiently. With the entire experience of a world-wide successful enterprise.

| | |
|---|---|
| Telephone | 07022/9660-132, -231, -230 |
| eMail | support@systeme-lauer .de |
| Web site | www.lauer-systeme.net |
| | Systeme Lauer Active Area |
| | (Download of Software, driver, manuals, Forum...) |

# Safety regulations

This reference manual contains the most important remarks in order to safely operate the device.

- This operator,s guide, particulary the safety remarks are to be noted byall persons working with the device.
- Furtherrmore, the rules and regulations for the accident prevention applying to the application location are to be observed.
- Use as directed. The device is deigned for the application in the industrial area.
- The device is manufactured to the state of the art and the official safeguarding regulations. Nevertheless, due to the application, dangers or impairments can result to the machine or to material assets.
- The device meets the requirement of the EMC guidelines and harmonized European standards. Any hardware-related modification of the system can influence the EMC behavior.
- The device may not be used without special protective measures in the hazardous area and in plants requiring a special monitoring.
- Do not heat up the buffer batteries. Danger of explosion. Serious burning can be the result.
- The installation and operation may only be performed by trained personnel.
- The operating voltage of the device may only be in the specified ranges.
- You find information on this on the type plate and in the specifications of this reference manual.

# Norms

The device is constructed using up-to-date technologies and fulfils the requirements of the following guidelines and norms:

- Compliant with the EMC Directive 89/336/EEC and the German law on electro-magnetic compatibility
- Interference compliant with the generic requirements norm EN 50081-2 and product norm EN 55022:
- Measurement of the conducted interference voltage as per EN 55022
- Measurement of the radiated radio interference field power as per EN 55022 class A
- Interference immunity in compliance with generic requirements norm EN 50082-2 and product norm EN 61000-6-2:
  - Electro-static discharge (ESD) as per with EN 61000-4-2
  - High-frequency electromagnetic fields as per EN 61000-4-3 and ENV 50204
  - Fast transient interference (burst) as per EN 61000-4-4
  - Surge voltages as per EN 61000-4-5
  - High-frequency conducted fields as per EN 61000-4-6
  - Voltage dips and short-term interruptions as per EN 61000-4-11

The assembly and connection instructions contained in this documentation must be followed.

Conformity of this equipment is confirmed by the CE logo.
The EC declaration of conformity can be requested from:

Systeme Lauer GmbH & Co KG
P-O-Box 1465
D-72604 Nürtingen

# A PCS Direct driver

## A1 First commissioning

Introduction

The first contact and the commissioning of control equipment will be very easy for you. Featuring the easy to understand and practice-related menu system, the PCSPRO/PCSPRO$^{WIN}$ configuration software guides you quickly to your objective. The integrated extensive help system assists you if you are not familiar with individual terms. Using F1, you find answers to your questions at any time. Our mailbox or our hotline are available to you in difficult cases.

About what this reference manual reports

In the following, the use of a Lauer PCS micro/mini display with a Siemens S7-CPU214 programmable controller is described. It is explained how the two systems are to be connected and how to perform a commissioning. The asynchronous or the synchronous operation and the pertinent handling module are explained in detail.

Necessary devices and accessories

The following products are required for the operation of a programmable controller with an already parameterized PCS micro/mini of Systeme Lauer.

- The PCS operating console itself (already parameterized)
- PCS721 adapter cable for connecting the PCS to the programmable controller using the RS-485 interface
- PCSPRO/PCSPRO$^{WIN}$ floppy disk with S7driver and the PCSPRO/PCSPRO$^{WIN}$ reference manual

Furthermore, the following items are required from the Siemens company.

- One S7 CPU200 controller
- S7 TOOLITE programming software
- A PPI cable for programming the programmable controller

... and power supplies for all components

Method of procedure

1. Creating a project with PCSPRO/PCSPRO$^{WIN}$.
2. Connect the PCS operating console to the PC.
3. Transfer the project from the PC to the PCS.
4. Connect the PCS to the programmble controller.

Connection of the PCS to the PLC

1. At the rear of the PCS, set DIL switches 8 and 9 to ”OFF”.
2. Set DIL switches 5 and 6 of the PCS to ON.
3. Apply 24V (19 .. 33 V) operating voltage to the PCS.
4. At least, the ERR LED must light now.
5. Connect the programmable controller and the PCS by using the PCS 721 cable.
6. Switch the programmable controller from Stop to Run.
7. At the PCS, the ERR LED must be deactivated now.

# A PCS Direct driver

## A2 Driver

Loading of the driver into the PCS

Project data and selected drivers are transferred when loading data into the PCS. The driver for the S7 is called ”SIES72PS.DRV”. In PCSPRO/ PCSPRO$^{WIN}$ or , select the Transfer menu item and save your project in the PCS. You can read the detailed method of procedure in PCSPRO/ PCSPRO$^{WIN}$ or in the reference manual.

Driver setting values

By default, the variables are pre-assigned; usually, no change is required. If one is required then the settings can be performed using PCSPRO/ PCSPRO$^{WIN}$ or the ”Driver” menu item. For the S7driver, the following variables can be set individually.

Variable AA

Time-out time
The time-out time defines the max. permitted time between the data packages. By default, this time is 400. This corresponds to 4000ms. For variable AA, values from 200 to 9990 are permitted
(= 2 to 9.99s).

Variables AC, AD, AE, AF

Interface and Mode of operation
DIL switches 5 and 6 at the rear of the PCS allow 4 possible settings.

| | | **DIL 5** | **DIL 6** |
|---|---|---|---|
| „NO SYNC, | RS485“ | OFF | OFF |
| „SYNC, | RS485“ | ON | OFF |
| „NO SYNC, | RS232“ | OFF | ON |
| „SYNC, | RS232“ | ON | ON |

”SYNC ”defines the use and processing of a synchronization word in the PLC, ”NO SYNC ”uses no synchronization word.
Different settings can be assigned to each of the 4 DIL switches.

Variable AL

Start address word
Defines the first word of the data area to be used. The variable has the function of an offset. Possible values are from 0 to 4064, e.g. AL = 10 allocates PCS word 0 to VW10 (= 10th word of the variables area). Default value is 0.

Variable AM

End address word
Defines the last word of the data area to be used.
Possible values are 30 to 4094, e.g. AM = 100 limits the data words to be used to VW100 max.
Thus, the permitted data area is AM-AL+1 = x words in size.

Variable AH

PCS station number
Defines the address of the PCS. Set to 1 by default since communications is always initiated in the PCS. Thus, the PCS is the master.

# A PCS Direct driver

Variable AO

Programmable controller station number
Defines the address of the programmable controller. Set to 2 by default since communications is never initiated by the programmable controller. Thus, the programmable controller is the slave.

For the programmable controller and PCS access, a common data area has to be defined. In the programmable controller, this area must be physically available and defined, e.g. the variables area of the CPU214 is 4096 MW max. in size.

In the PCS, this area is defined by using the AL and AT driver variables. AL defines the start word. With this, AL = 4 e.g. defines the start value of a data area at VW4 (4th word of the variables area). AM specifies the last word to be used, e.g. AM = 200 sets the last word to be used to VW200 (word 200).

## A2.1 Effective PCS/PLC response times

The NoSync operation is faster than the Sync operation. In the NoSync operation, the time requirement is approx. 60+ 30*n words in ms. This is the most frequent error at the first commissioning and during the continuous operation:

**PCS diagnosis is set**
DIL switch no. 8 is set to ON. After turning on, the PCS enters a diagnosis routine that is only needed for test purposes if this switch is set.

Remedy:
Set DIL switch 8 to OFF and re-start the PCS (by cycling power or by briefly pressing the RESET pushbutton above the DIL switches).

# A PCS Direct driver

## A2.2 Error messages of PLC

Time-out

A time-out is reported in the programmable controller

- In this instance search for an error in the programmable controller « PCS connection
- Possibly, the cable is faultily or incorrectly plugged in
- A wrong driver has been loaded into the PCS or an interface is used with one of the PCS DIL switches in a wrong position

Progr. contr. in Stop

After the start, the programmable controller switches to Stop

- Diagnosis: Data area in programmable controller is not available or incorrect handling software is used

PCS error messages

- Communications starts (PCS ERR LED is deactivated) but after a certain time, the following message is displayed on the upper display line of the PCS: »COMMUNICATION ERROR«.

## A2.3 PCS Diagnosis

- A helpfull PCS diagnosis is the output of the PCS status on the display
- After a PCS start or a PCS reset, the ERR LED lights statically since the PCS is still off-line
- Press the Help key
- Using the Help plus arrow keys, you can display the PCS version, the data record and the driver version as well as driver variables.

# A PCS Direct driver

Installation tips

- Connect the cable screening to the central grounding point of the control cabinet.
- Provide for good ground connections to the PCS housing on the one hand and to the programmable controller bus board on the other hand.
- Consider that based on its large surface, a homogeneous copper earthing strap features a considerably better RF conductivity as a normal switching wire.
- Avoid as much as possible the generation of high-frequency interference as these are very difficult to be dampened. Potential separation exists between the programmable controller and PCS by means of optocouplers; but in case of fast transients this potential separation is ineffectively as optocouplers possess a coupling capacity (even only significantly small).
- Provide for clear reference points of the supply voltages. For this, the power supply unit is potential free (floating).
- With interference-loaded supply voltage, the use of its own power supply unit is advisable. It should possess corresponding interference filters. Using the ground wire, 0 volts can then directly be connected to the PCS.
- The PCS and the adapter cable should possess a minimum clearance of 200 mm to interference sources. This applies specifically to inductivities and frequency converters.
- Make sure, that the serial data lines are covered by screenings as completely as possible.
- On the PCS and on the programmable controller side use a metalized plug housing that is well connected to the cable screen.
- Make sure that with grounding on both sides a potential equalization line of at least the 10-fold shield cross section is sometimes required. Particularly if PCS and programmable controller are connected to the same earthing point (e. g., PCS and programmable controller are located in the same control cabinet).
- **Reason:** To prevent equalizing currents on the cable screen. The S7 protocol frame is extensive so this burdens the throughput. Thus, it is slower than the AS511 protocol for example.

# A PCS Direct driver

## A2.4 SYNC or NOSYNC

Since an asynchronous data access takes place in the programmable controller cycle, the data written by the PCS can be overwritten by the programmable controller and reversed. Therefore, data consistency is not provided. A remedy against this problem is the use of a synchronization word: SYNC operation. After the complete transfer from the PCS, the programmable controller receives a signal that the data are released now. These data are now available to the programmable controller. After processing, the programmable controller signals that new data can be transmitted now. The programmable controller sets the left byte of DW3 to 0 and inverts the right byte of VW3. The data exchange starts again. With the Sync operation, time-out timers should monitor this mode of operation on both sides.
The Sync operation requires a handling module in the programmable controller.
The NoSync operation is faster than the Sync operation.

Asynchronous PLC to PCS operation

Actual data words and preset data words have strictly to be separated (writing accesses could interfere with another). Even the reading of e.g., data words spread across several words can cause an error; this happens at the time a variable is read although only a part of the variable has been written.
Bit variables should only be used 1 time per word as the access of the PCS only takes place on a word basis. A word fetched from the PCS that is modified and written back can overwrite another bit variable on this word.
The same applies to the message bits with erase behavior 2 (PCS resets the message bits). Therefore, avoid erase behavior 2 or use only 1 message per message word.

**Advantages compared to the SYNC operation are**

- faster data exchange. The cyclic time of the programmable controller does not influence the response time.
- No additional handling module is required for the communication. Only the data area must be available in the correct size. At any time, accessing the data area in the programmable controller program is possible.

Synchronous PCS to PLC operation

The access to the data in the programmable controller has to be synchronized if you want to use the entire functional range of the PCS, i.e. programmable controller and PCS access the data alternately. In addition, a synchronization word is transferred to the programmable controller. An handling module checks this word and enables access to the programmable controller user program. The synchronization word is modified and the PCS accesses the data area if the user program has finished processing the data words.
While the PCS processes the data, the user program may not access the data. This Ping Pong play game enables also a time-out monitoring on the PCS side. The timer is re-started every time the PCS reads the inverted synchronization word. A time-out exists if the timer runs down.

# A PCS Direct driver

By mutual access, actual values and preset values can be mixed, bit variables can be used, and erase behavior 2 can be implemented, etc. Therefore, the entire intelligence of the PCS is available to you. The disadvantage is that the reaction speed between PCS and programmable controller is decreased.
Furthermore, the programmable controller program must inquire whether accessing any data is permitted.
The time-out time, i.e. the time which passes from the interrupt of the communication up to the signaling in the programmable controller should be set to min. 2 seconds. In the PCS, the time-out time is set via the ”AA ” driver variable.

## A2.5 Handling software in the SYNC operation

In order to realize synchronous communication between PCS and programmable controller,

- the ”SYNC ” setting at the PCS must be selected
- a small handling module must be linked in at the programmable controller

In the following, an example of this programmable controller handling module is described. Obviously, you can solve these tasks differently in your software; it is only important that you adhere to the following sequence.

1. The programmable controller initializes VW3 using ”FF, 00 ”.
2. The PCS places a job number in word 3 (each time incremented by 1), e.g.: ”01,01 ”. This is the signal for the programmable controller that the data area is released. Furthermore, the timeout timer in the programmable controller is re-started.
3. VW3 is set to ”00, FE ” if the processing of the data area in the programmable controller is finished. The package number in the right byte is inverted. For the PCS, the read inverted package number is the enable for the renewed data communication. In the data area, nothing may be changed anymore by the programmable controller program.

Now, step 2 and 3 are cyclically repeated. The process is resumed with step 1if a time-out occurs.

## A2.6 Example of synchronous handling software

An S7-CPU 214 is selected for the following example. It is assumed that synchronous communication takes place. The area VW0 to VW255 is used. Therefore, the driver variables used in the PCS must be:
AA=400, AC, AD, AE, or AF= ”SYNC , RS-485", AL=0, AM=255.

In the example, T2 is used for time-out monitoring with a time set to 4 seconds; flag bit 20.0 is used for controlling the T2 timer. Output 1.0 is set if T2 expires. If output 0.0 is = 1, communications is restarted after a time-out (DW3 is processed again); with output 0.0 = 0, communications is not restarted (DW3 is not processed).

# A PCS Direct driver

Step 1

Initalization
Pre-assignment and communication start

```
INIT
MOVW    0,VW8          clear PCS status
MOVW    0,VW10
MOVW    0,VW12
MOVW    0,VW14
MOVW    0,VW16
MOVW    0,VW18
MOVW    0,VW28

MOVW    4040,VW26      transfer of all data release words
MOVW    ff00,VW0       communications reset
MOVW    ff00,VW0       last job number reset
MOVW    ff00,VW1       no processing in FB213
MOVW    ff00,VW3       sync word
```

Step 2

Communication processing
The job number is tested and calculated and time-out is monitored. Output 0.0 is set if a time-out occurs.

S7 sync operation with pcs90, pcs900

```
network
ld              sm0.1           // first run
movw        255,vw4         // dw2 = 0x00ff
movw        255,vw6         // dw3 = 0x00ff
lbl   1

network
ld              sm0.0
movw        4040,vw26       // enable modus
ld              sm0.0
movb        vb8,vb20        // copy functionkey to led
movb        vb20,qb0        // show     functionkey to plc-port
movw        0,vw22
ld              sm0.0
ton             63, 40              // 40 * 100ms

ld              t63             // no timeOut ?
not
ab=             vb6,vb7         // compare
invw            vw6             // complement jobNr
movb        0,vb6
movb        vb7,vb2
movb        1,qb0
r               t63,1
lbl   4

ld              t63             // time-out
=I              q0.0            // signal time-out
ld              t63             // time-out is set
a               i0.0            // only reset with timer3
momentary pushbutton
r               t63,1

network
mend
```

# A PCS Direct driver

## A2.7 Speed optimization

Data transmission speed mainly depends on two facts.

1. the enabled transmission functions in the command words
2. the number of variables on the shown display page. The following measures can be taken to accelerate the transmission of data.

**PLC program optimizations for the PCS 009/090/095**
Disable all not required functions in the command words via the programmable controller program. By this means, the transmission requirement of constantly transmitted data is decreased. For this and using bits 0..3 of DW13, you can limit the number of message words in the variable word 13, command word A . It is sufficient to read 3 words of message bits if you only need 35 messages for example. This can be set by writing xxxxxxxx xxxx0011 to DW13. According to the requirement, this setting can be (dynamically) changed by the programmable controller at any time.

- By setting bit 7 of DW13 to a logical 0, you block the reading (transfer) of all W10. . 11 LED STATUS WORDS in case of the PCS 009/090 and W24. . 25 in case of PCS 095.
- By setting bit 6 of DW13 to a logical 0, you block the reading (transfer) of the display and storage behavior.
- Avoid frequent changes of the display text as with a change, the status words 6 to 9 are also transferred.
- You can dynamically change the transferred volume of data within your programmable controller program.
- Proceed as follows to implement a jog operation for example.
- Disable all functions as described above. Call jog operating text without variables. After ending the jog operation, the transmission functions are released again.

**PCSPRO/PCSPRO$^{WIN}$ program optimizations in case of PCS 009/090/095**

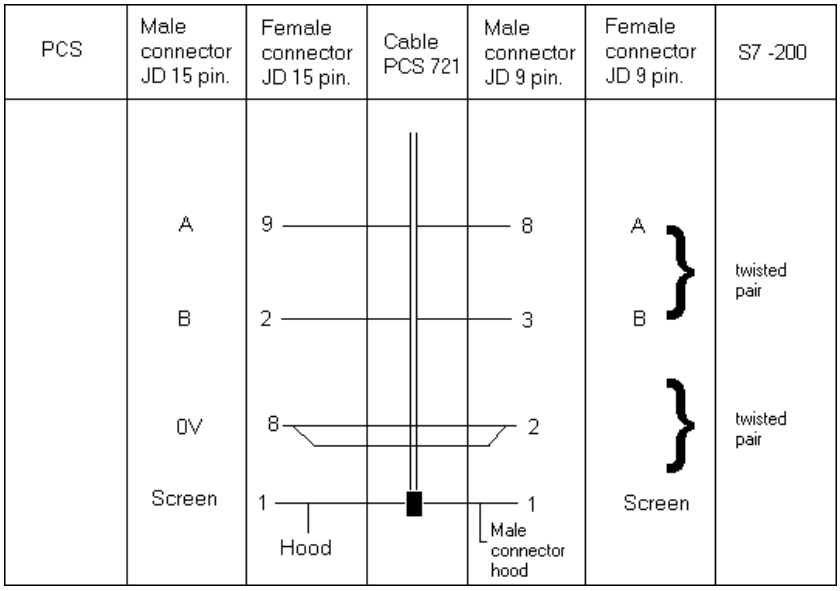
- Include as few variables as possible on the shown display page because the transferred data volume data increases with the number of variables.
- If several variables should be shown on the same display page, it is advantageous to address them consecutively.
- Then, several variables can be transferred in one write or read command and the transmission speed increases.
- If for example the first variable on the display is located at DW50 then the further variables should be located at data words 51, 52, 53... etc.

# A PCS Direct driver

## A3 Communication

PCS 721 Adapter cable

PCS/PLC connection via the RS-485 interface




By using a shielded normal cable (4 * 0.14, not twisted) a maximum length of 20 meters is recommended.
The 10-fold length can be realized by using data cables with low capacity twisted in pairs!
Recommended cable: 2 * 2 * 0.2² stranded in pairs with individual shielding.

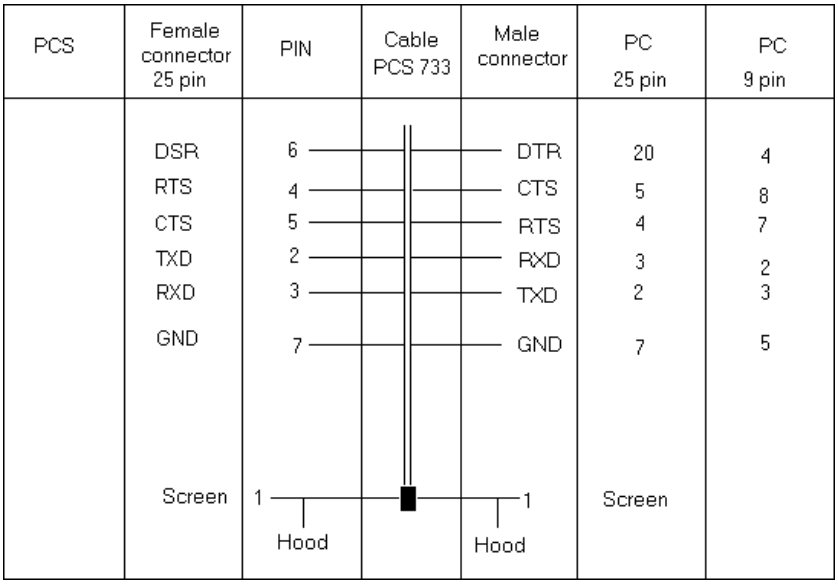
**Screening of the adapter cables**
With a metalized plug housing, the screening should be connected at both ends. When using non-metalized plug housings, the screening can also be connected to pin 1; nevertheless, this is not to be recommended for noise reasons as the data lines should be covered as completely as possible by the screening.
However, by grounding both sides it is to be noted that possibly (because of ground potential offsets) a potential equalization line of at least the 10-fold cross section of the screening is required.

**Reason:** Adjustment currents should not flow via the cable screening. Particularly, if PCS and programmable controller are not connected to the same earthing point. For example, this is the case if PCS and programmable controller are not accommodated in the same control cabinet.

# A PCS Direct driver

PCS 733 Programming cable

PC/PG 730 and 750-PCS connection for transferring project data from the PC to the PCS



# A PCS Direct driver

# B PCS topline midi & PCS plus/win Expander driver

## B1 First commissioning

Delimitation

The successful parameterizing of the PCS midi is assumed. This reference manual refers to the exclusive use of the PCS micro/midi/mini in connection with an S7 CPU200 controller of the Siemens company.
In the following, this controller is named as programmable controller and the programming software for the programmable controller as TOOLITE. The Siemens specific terms and the programming of the programmable controller using TOOLITE are assumed to be known. This driver was developed on the S7 CPU200 controller. The PCS is connected to the programming interface of the controller.

Attention!
Use only the PCSPRO or PCSPRO$^{WIN}$ software for configuring. Other software packages can trigger malfunctions in the PCS and in the programmable controller.

Necessary devices and accessories

The following products are required for the operation of a programmable controller with an already parameterized PCS micro/mini of Systeme Lauer.

- The PCS operating console itself (already parameterized).
- The PCS 733 programming cable for the PC/PCS connection via RS-232.
- The PCS 721 adapter cable for the PCS/programmable controller connection via RS-485.
- This reference manual.
- PCSPRO\PCSPRO$^{WIN}$ floppy disk and reference manual.
- SIEMENS Master floppy disk with P900_2_0.AWL and P09_2_0.AWL handling software for the S7 CPU 200.

Furthermore, the following items are required from the Siemens company.

- one S7 CPU200 controller.
- S7 TOOLITE programming software.
- A PPI cable for programming the programmable controller.

... and power supplies for all components.

# B PCS topline midi & PCS plus/win Expander driver

Loading of the operating software

Note!
Check the function of the handling software in order to avoid malfunctions in the PCS or in the programmable controller.

1. Connect the programmable controller and the PC using the PLC programming cable
2. Start the TOOLITE software on the PC
3. Load the P900 program_2_0.AWL or P090_2_0.AWL
4. Transfer the program to the programmable controller

Now, you can connect the PCS as described in section B2.3.

# B PCS topline midi & PCS plus/win Expander driver

## B2 Driver

Loading of the driver into the PCS

When configuring the PCS, both the user program with data and the selected driver will be transferred. In this case, the driver is called ”SIES72FP.DRV”
For configuring a PCS midi, set DIL 7 to the corresponding baud rate ( ”OFF ”= 38.5 Kbaud, ”ON ”= 115 Kbaud), DIL 8 to ”OFF ”and DIL 9 to ”ON ”. Connect PCS and PC using the PCS 733 programming cable. The SIES72FP.DRV driver is automatically loaded after selecting the expander driver for the S7 in the PCSPRO or PCSPRO$^{WIN}$programming software. By default, an offset = 0 is defined; therefore VW0 is used as start value. If other values are used as default values, the variables in the SIES72FP.DRV programmable controller handling software also have to be shifted. This is described in chapter 3 in more detail.

Driver variables of the PCS midi

In the PCSPRO/PCSPRO$^{WIN}$ software, the variables for the SIES72FP. DRV driver can be set under the ”Project/Driver Parameter ”menu item. The contents of the variable can be shown using the offline menu of the PCS.

PCS time-out

The time-out time defines the maximum admissible time for the job processing in the programmable controller. By default, this time is 4000. This corresponds 4000 ms = 4.0 seconds. Values from 0 to 9990 are permitted, corresponding to 0 to 9.99 seconds.

Baud rate and transmission type

The baud rate and the interface, the PCS and the programmable controller should communicate with are set using the DIL switches 5 and 6 of the PCS. You can select between the settings that have been made in PCSPRO under the ”Driver variables” menu item. The default settings are represented in the table below.

**PCS midi**

| DIL5 | DIL6 | Variables | preset values |
|---|---|---|---|
| off | off | AC | 9600 bauds RS485 |
| on | off | AD | 9600 bauds RS232 |
| off | on | AE | 9600 bauds RS485 |
| on | on | AF | 9600 bauds RS232 |

**PCS micro/midi**

| DIL5 | DIL6 | Variables | preset values |
|---|---|---|---|
| off | off | AC | 19200 bauds RS485 |
| on | off | AD | 9600 bauds RS485 |
| off | on | AE | 19200 bauds RS485 |
| on | on | AF | 9600 bauds RS485 |

# B PCS topline midi & PCS plus/win Expander driver

Tasks per package

Defines the number of sub-packages for the data exchange. The default setting is AJ = 10. If AJ is reduced, the transmission time of tasks with high priority is reduced, e. g. key tasks. Tasks with lower priority e. g. actual values) are accordingly transferred less often. If AJ is increased, variables are refreshed more quickly but the key transmissions last somewhat longer. AJ is settable between 1 and 20.

Connection of the PCS to the PLC

- PCS is configured and loaded with the project.
- The P900 _2_0.AWL or P090_2_0.AWL expander module is loaded into the programmable controller.
- After configuring the PCS, set the DIL switches and 9 at the rear of the PCS to ”OFF”.
- Apply 24V (19 .. 33v) operating voltage to the PCS Now, the ERR LED must light at least.
- Connect the programming interface of the programmable controller to the PCS using the PCS 721 adapter cable.
- Set the programmable controller to ”RUN ”.
- Now, the ERR LED at the PCS must be deactivated. Idle text 0 is shown on the display of the PCS. Continue reading in section 2.5 if this is not the case.

Warning!
Check the function of the PCS after parameterizing or driver installation. All parameterized functions have to be tested. Otherwise, malfunctions of the PCS or programmable controller are possible.

Effective PCS to PLC reaction times

The response time of the protocol depends heavily on the tasks in the PCS. If variables are displayed or even processed, the communication cycle time is higher compared to a text without variables. Even the transmission of the message bit area and the LED words is significant. Limit these transfers to the ones really necessary. You can do this also in the running operation, e. g. to implement key jog operation. Also, the response time significantly depends on the cyclic time of the programmable controller since the programmable controller serves the communication at the end of a cycle.

The speed of the communication can be seen at the so-called ”Key → LED ”Time, therefore the time in which a key is transferred to the programmable controller and an LED is set in the PCS. This time consists of 2 communication cycles and a programmable controller scan cycle. The response time, i.e. the time in which a key is reported in the programmable controller is half the amount.
For a fast communication, make sure that the variables are stored consecutively in the data word area. Then, several variables can be treated using one write or read command.
Communications between PLC and PCS are constantly monitored by both participants. Error indication in the PLC is performed by setting the Q 0.0 error output. In case of communication errors, a corresponding error text is shown on the display of the PCS and the ERR LED flashes.

# B PCS topline midi & PCS plus/win Expander driver

**Notes for connecting the PCS to a PLC**

- Check the function of the PCS after parameterizing or driver installation. All parameterized functions have to be tested;, otherwise, malfunctions of the PCS or PLC are possible.
- Connect the cable screening to the central grounding point of the control cabinet
- Provide for good ground connections to the PCS housing on the one hand and to the programmable controller bus board on the other hand. Note that based on its large surface a homogeneous copper earthing strap features a considerably better RF conductivity than a normal switching wire
- As much as possible, avoid the generation of high-frequency interference as these are very difficult to be dampened. Potential separation exists between the programmable controller and PCS by means of optocouplers; but in case of fast transients this potential separation is ineffectively as optocouplers possess a coupling capacity (even only significantly small) .
- Provide for clear reference points of the supply voltages. For this, the power supply unit is potential free (floating).
- With noisy supply voltage, the use of an own power supply unit for the PCS (24 volts, 10 VA) is advisable. It should possess corresponding interference filters. Using the ground wire, 0 volts can then be directly connected to the PCS.
- The PCS and the adapter cable should possess a minimum distance of 200 mm to interference sources. This specially affects inductivities and frequency converters. Make sure, that the serial data lines are covered by screenings as completely as possible. On the PCS and on the programmable controller side use a metalized plug housing that is well connected to the cable screen.
- Make sure that with grounding on both sides a potential equalization line of at least the 10-fold shield cross section is sometimes required. Particularly, if PCS and programmable controller are connected to the same earthing point (e. g., PCS and programmable controller are located in different control cabinets).
- **Reason:** Equalizing currents on the cable screening should be avoided.
- The used SIES72FP driver is a expander driver, i.e. it exchanges the data area between PLC and PCS via task packages. An PLC program is needed for that this. PLC and PCS communicate using a Free Port protocol via RS-485 with 9600 bauds, (PCS micro/mini selectively with 9600/19200 bauds) 8 bits, even parity and 1 stop bit.

# B PCS topline midi & PCS plus/win Expander driver

## B3 PLC handling software

P900_2_0.AWL expander module

The expander module is apportioned into the partial blocks defined below.

Start up

This part is active only once after starting the PLC.

- initialize the PCS
- set interface to a new protocol
- set receive interrupt
- supply parameters
- install monitoring timer

Receive interrupt

All characters of the PCS are received here and are stored until they are evaluated.

- receive character and store it in the receive buffer
- monitor receive length

Job evaluation

The data transmitted by the PCS are checked and evaluated. If the PCS requests data, a corresponding data package is assembled and sent to the PCS. After processing of all jobs of the PCS, the PCS may now send new jobs.

- the jobs transmitted by the PCS are evaluated and processed, necessary answers are returned to the PCS, monitoring timers are retriggert.

PLC time-out

A monitoring function is implemented in the PLC that monitors the proper operation of the entire system. With every transmission, a transmission number is written into the programmable controller. This number is constantly incremented by 1. This results in a constant toggeling of the lower bit of the job number. A programmable controller timer is thus retriggered. With the missing of a new job number, the monitoring timer in the programmable controller is activated and sets output bit 0.0.
If this case occurs, the key bits should be set to zero by the programmable controller.

COM_ERR (Q0.0): Output that is active on a communication failure. The communication can be restarted after a failure using the I.0.0 restart input.

Attention!
The interface of the programmable controller is set anew in the first start up of the programmable controller. Then, no PPI protocol is possible anymore. Switch the programmable controller to STOP to make the PLC accessible to the TOOLITE programming software again. Now, access is possible again using TOOLITE.
By default, the address area in the PLC is located at address VB0. If your application requires another address, the ”&VB0 ”command must be replaced by your desired starting address.

P090_2_0.AWL expander module

The expander module is apportioned into the partial blocks defined below.

# B PCS topline midi & PCS plus/win Expander driver

PLC START UP

This part is active only once after starting the PLC.

- initialize the PCS
- set interface to a new protocol
- set receive interrupt
- supply parameters
- setup monitoring timer

FILL RX-BUFFER

All characters of the PCS arrive here and are stored until evaluated.

- receive characters and store them in the receive buffer
- monitor the receive length

SEND TX-BUFFER

The data transmitted by the PCS are checked and evaluated. If the PCS request data, a corresponding data package is assembled and sent to the PCS. After processing of all jobs of the PCS, the PCS may now send new jobs.

- the jobs transmitted by the PCS are evaluated and are processed, necessary answers are returned to the PCS, and monitoring timers retriggered

PLC TIME-OUT

A monitoring function is implemented in the PLC that monitors the proper operation of the entire system. With every transmission, a transmission number is written into the programmable controller. This number is constantly incremented by 1. This results in a constant toggeling of the lower bit of the job number. A programmable controller timer is thus retriggered. With the missing of a new job number, the monitoring timer in the programmable controller is activated and sets output bit 0.0.
If this case occurs, the key bits should be set to zero by the programmable controller.

COM_ERR (Q0.0): Output that is active in case of a communication failure or a block check error.

Interface setting

The transmission rate is transferred to the SMB30 system flag. Default setting is 69 decimal (19200,8,even,1). For 9600 bauds, 73 decimal is entered.

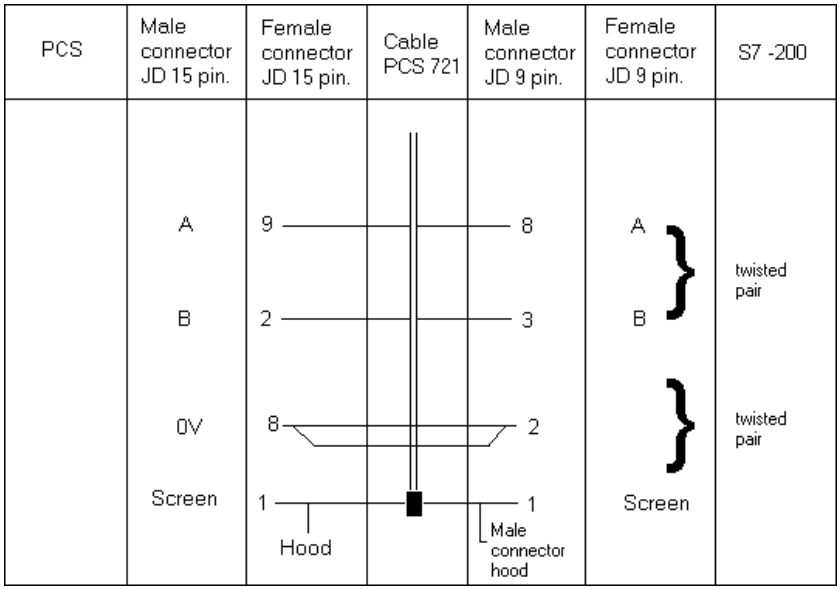
Attention!
The interface of the programmable controller is set anew in the first start up of the programmable controller. Then, no PPI protocol is possible anymore. Switch the programmable controller to STOP to make the PLC accessible to the TOOLITE programming software again. Now, access is possible again using TOOLITE.
By default, the address area in the PLC start at address VB0. If your application requires another address, the ”&VB0 ”command must be replaced by your desired starting address.
Note the memory storage by the FP protocol.

# B PCS topline midi & PCS plus/win Expander driver

## B5 Communication

PCS 721 Adapter cable

PCS - S7 CPU 200 connection via the RS-485 interface



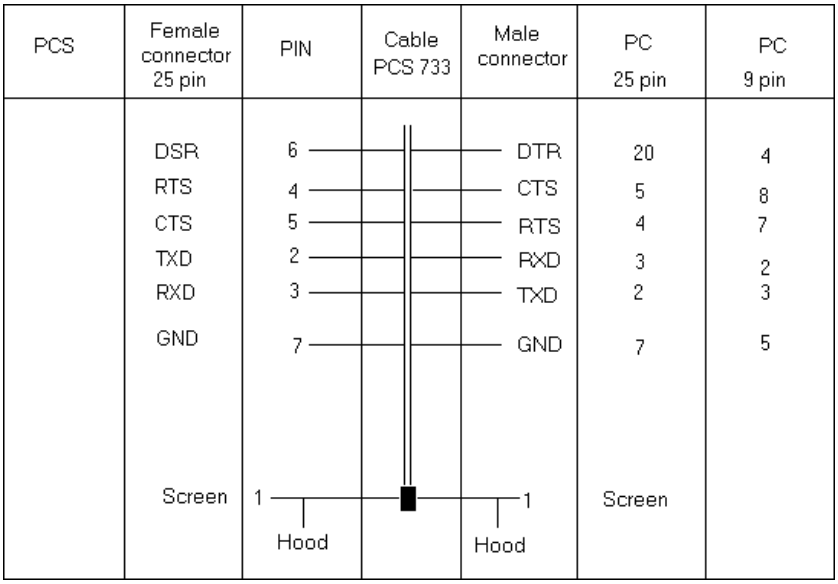

**Screening of the adapter cables**
On both sides, the screen should be connected to a metalized plug case. When using non-metalized plug housings, the screening can also be connected to pin 1; nevertheless, this is not recommended for noise reasons as the data lines should be covered as completely as possible by the screening. However, by grounding both sides it is to be noted that possibly (because of ground potential offsets) a potential equalization wire of at least the 10-fold cross section of the screening is required.

**Reason:**Equalizing currents should not flow via the cable screening. Particularly, if PCS and PLC are not connected to the same earthing point. For example, this is the case if PCS and programmable controller are not accommodated in the same control cabinet.

# B PCS topline midi & PCS plus/win Expander driver

PCS 733 programming cable

PCS - PC connection




PCS - PLC data transfer

The data traffic with the controller is performed using data packages. Every data package is provided with a check sum. Its contents is checked on possible errors by the PLC and the PCS. In addition, every communication cycle is provided with a continuous job number.

The PCS is the communication master. It is the task of the PCS to set up the communication and to send jobs to the PLC. PCS and PLC communicate asynchronously via RS-485 using the serial interface. The transmission rate is settable to 9600. Eight data bits, even parity, 1 stop bit are firmly set for communications via the PU interface.

Only the schematic exchange of the data is described in this reference manual.

Structure of the read cycle

| **PCS** | | **PLC** |
|---|---|---|
| READ N BYTES | → | |
| | ← | ANSWER N BYTES |
| (Repetition in accordance with data field) | | |
| READ N BYTES | → | |
| | ← | ANSWER N BYTES |

Structure of the write cycle

| **PCS** | | **PLC** |
|---|---|---|
| WRITE N BYTES | → | |
| | ← | ( ERROR ) |
| (Repetition in accordance with data field) | | |
| WRITE N BYTES | → | |
| | ← | ( ERROR ) |

# B PCS topline midi & PCS plus/win Expander driver

## B5.1 Speed optimization

The data transmission speed depends mainly on two facts.

- The enabled transmission functions in the command words
- The number of the variables represented on the shown display page

By a bad organization of the transmission, you are able to multiply the transfer times. The following measures can be taken to accelerate the transmission of data.

- Using the driver variable AJ you can change the refresh behavior of the PCS tasks. A small AJ enables short communication cycles and, therefore, the fast exchange of key data; however, variables last longer.
- One large AJ packs many jobs into one communication cycle and therefore causes the fast refreshing of variable; however, key transmissions take longer.
- Note that in case of a small AJ number, the key LED check does not optimally functions at a short depressing of the key since the key´s erase action has a higher priority than reading the LEDs.

**PLC program optimizations for the PCS 009/090/095**

- Disable all unnecessary functions in the command words using the PLC program. By this means, the response time of constantly transmitted data is decreased.
- Using bits 0..3 of DW13, you can limit the number of message words in data word 13, command word A.
- It is sufficient to read 3 words of message bits if you need only 35 messages for example. This can be set by writing xxxxxxxx xxxx0011 to DW13. According to the requirement, this setting can be (dynamically) changed by the programmable controller at any time.
- By setting bit 7 of DW13 to a logical 0, you disable the reading (transfer) of all W10 .. 11 LED STATUS WORDS in case of the PCS 009/090 and W24 .. 25 in case of PCS 095
- By setting bit 6 of DW13 to a logical 0, you disable the reading (transfer) of the display and storage behavior
- Avoid frequent changes of the display text as with a change the status words 6 to 9 are also transferred
- You can dynamically change the transferred volume of data with your programmable controller program
- Proceed as follows to implement a jog operation for example
- Disable all functions as described above.
- Call jog operating text without variables.
- Re-enable the transfer function after completing of the jog operation.

# B PCS topline midi & PCS plus/win Expander driver

**PLC program optimizations for the PCS 900/920/950**

- Disable all unnecessary functions in the command words using the PLC program. By this means, the response time of constantly transmitted data is decreased.
- Specially, keep attention to the transmission of time and message words
- In addition, using data word 37 - command word B, you are able to limit the number of the message words via bits 0 .. 7
- For example, it is sufficient to read a message block containing 8 words if you need less than 128 messages. This can be set by writing xxxxxxxx 00000001 to DW37. According to the requirement, this setting can be (dynamically) changed by the programmable controller at any time.
- By setting bit 4 of DW36 to a logical 0, you disable the reading (transfer) of all LED STATUS WORDS W20. . 27
- By setting bit 7 of DW36 to a logical 0, you disable the reading of command words C, D and E
- By setting bit 5 of DW to a logical 0, you disable the transfer of the clock. This is specially important as the clock is transferred every second and therefore heavily loads the communication. Therefore,, only release the clock transmission if it is absolutely needed.
- By setting bit 6 of DW36 to a logical 0, you disable the transmission of the date. This has only a small meaning since the date is only transferred when changed therefore once a day.
- Avoid frequent changes of the display text as with a change the status words 6 to 9 are also transferred
- You can dynamically change the transferred volume of data with your programmable controller program
- Proceed as follows to implement a jog operation for example
- Disable all functions as described above.
- Call the jog operating text without variables.
- After ending the jog operation, the transmission functions are released again.
- In order to relieve the PLC program, you can use the softkey functions to switch idle texts, menus ect. You can always block this option by redefining the softkey bar using the PLC program.

# B PCS topline midi & PCS plus/win Expander driver

**PCSPRO \ PCSPRO$^{WIN}$ program optimization for the PCS 009/090/095**

- Include as few variables as possible on the shown display page because the transferred volume of data increases with the number of the variables.
- If several variables should be shown on the same display page it is advantageous to address them consecutively. Then, several variables can be transferred using one write or read command and the transmission speed increases.
- If for example the first variable on the display is located at DW50 then the further variables should be located at data words 51,52,53, , ... etc.

## B5.2 Communication error

For the PCS-PLC communication, the PCS works as a master and the PLC as a slave. Therefore, it is task of the PCS to setup the communication and to monitor it. For example, the following error message is output if errors occur during transmission.

**COMMUNICATION ERROR**
**TIME-OUT**

There is a time monitoring for the serial data exchange in the PCS and in the PLC. As default value, the time-out time is 4 seconds in the PCS. After that, an error message is displayed on the PCS and the ERROR LED flashes. In the background, the PCS attempts to reestablish the communication. The above shown error message is erased if this is successful. For example, the connector cable could be interrupted.

The PCS expects answer data requested from the PLC. No answers are returned to the PCS if the PLC is not connected or in STOP mode. The above-mentioned error message is displayed after several unsuccessful experiments. The error message is only erased after a successful data exchange.

The most frequent errors are listed here that are shown during the first commissioning.

**ERR LED of the PCS lights**

- DIL switch no. 8 is set to ON
- If this is set then after switching on, the PCS enters the diagnosis routine that is needed only for test purposes.

Remedy:
Switch off DIL switch 8 and re-start the PCS (by cycling power or briefly pressing the RESET pushbutton above the DIL switches on the rear panel).

# B PCS topline midi & PCS plus/win Expander driver

The ”Q0.0” error output in the PLC is set. This is the case if no new job is written into the PLC receive buffer within 2 seconds**.**

- Have you defined the same data area in the PLC and in the PCS
- Has the correct cable being used
- Is the cable faulty
- Set the I.0.0 restart input

The communication starts but however after a certain time, the message appears on the PCS

**COMMUNICATION ERROR**

- Have you defined the same data area in the PLC and in the PCS
- Has the correct cable being used
- Is the cable faulty
- The PCS/PLC connection is routed in a too noisy environment
- The earthing conditions are insufficient
- Is the PLC set to STOP mode
- Set the restart to input I.0.0.

Off-line menu

A helpful means of diagnosis for the PCS 009/090/095 is the outputting of the PCS status onto the display.

- Interrupt communication, press the Reset push-button
- Press the Help key now
- In addition, the ERR LED must light statically
- Using the Help and the arrow keys, the PCS version, the logical record version, the driver version and selected driver variables are displayed
- This option is no longer available if the communication is run successfully once

# B PCS topline midi & PCS plus/win Expander driver

## B6 PLC handling

Handling module for S7 CPU 212/214

```
// from  vb600 tx-buffer
// from vb650 rx-buffer
// 700 -719 pointer
// 720- 740 work bytes

// *******PLC START UP**************************************************
ld              sm0.1
movd  &vb650,vd704      // set rx-pointer
movd  &vb653,vd708      // set rx-work-pointer
movd  &vb0,vd712  // source data pointer (&vb0)
movb  0xff,vb651        //
movw  0,vw720           // rxCount
movw  0,vw722           // jobCount
movw  0,vw724           // byteCount
r           m6.0,3            // set step=0
movb  73,smb30          // 9600,8,e,1
atch        8,8               // set receiveIntProg to event 8
eni                           // enable interrupts
movw  0x0000,vw730
movd  0,vd600
movd  0,vd604

// ******* run ***********************************************************
ld          m6.0              // LCA jobs ?
call        10                // do the jobs, got from LCA
ld          m6.1
call        11                // send data to LCA
movb  vb1,qb0           // get LCA-keys
movb  vb1,vb4           // copy LCA-key to LCA-led: shows performance
MEND

network
// *******SEND TX-BUFFER***********************************************
sbr 11
ld          m6.1                    // start tx ?
a           sm4.5                   // tx free (running=0)
+i          3,vw724                 // txCount
movb  vb725,vb600       // 1.byte = txCounter for plc
movb  0x02,vb601              // STX = 1.TxByte
decw  vw724
movb  vb725,vb602       // 3.byte =2.TxByte=SendCounter for PCS
movw  0,vw726
movb  vb652,vb727       // get sendNo
xorw  0xffff,vw726
movb  vb727,vb603       // complement of sendNo in txBuffer
srw         vw724,1               // wordCounter
movw  0x0000,vw726
movd  &vb602,vd700            // set txPointer
lbl 1
xorw  *vd700,vw726            //  bild bcc
incd        vd700
incd        vd700
decw  vw724
ldw>= vw724,2
ld              sm0.0
movw  vw726,vw724
srw         vw724,8
movd  &vb600,vd700            // set txPointer
xmt         *vd700,0              // send on port 0,length=vb600,start at vb601
r           m6.0,4                // step=0
RET
```

# B PCS topline midi & PCS plus/win Expander driver

```
network
// *******DO THE JOBS************************************************
sbr 10
movw  0x0002,vw724            // tx-dataCount
movd  &vb604,vd700            // set tx-pointer data
lbl 0
ld          m6.0
//s         q0.0,1
//movb      vb723,qb0
movw  *vd708,vw730            // get actual job
andw  0x00f0,vw730            // what kind of job
movw  *vd708,vw732            // get actual job
andw  0x000f,vw732            // wordCount

movd  &vb0,vd712        // source data pointer ( &vb0: offset=0 )
movd  0,vd734                 // offset=0
movb  *vd708,vb737            // PLC wordAdr (vw734)
sld         vd734,1               // wordAdr
+d          vd734,vd712       // sourceAdr
decd  vd734

movw  *vd708,vw738
ldw=  vw730,0x0000            // no more jobs
jmp   61                            // loop done
ldw=  vw730,0x0010            // rd-job
call 12
ldw=  vw730,0x0020            // wr-job
call 13
ldw=  vw730,0x0040            // and-job
call 14
ldw=  vw730,0x0080            // or-job
ld          sm0.0
incw        vw722             // job count
incd        vd708             // next ABW
incd        vd708
//movw      *vd708,vw730            // get actual job
JMP 0                               // JOB-LOOP

lbl 61                              // all jobs  done
r           m6.0,1                  // step=1
movd  &vb650,vd704            // set rx-pointer
movd  &vb653,vd708            // set work-pointer
movd  &vb0,vd712        // reset source data pointer (&vb0:offset=0)
movw  0,vw722                 // jobCount
movw  0,vw720                 // rxCount=0
//movd      0,vd650
movd  0,vd654
movd  0,vd658
movb  0xff,vb651              //
ret

// ******* rd-job *********************************************************
sbr 12
ldw=  vw730,0x0010            // rd-job
s           m6.1,1                  // switch to tx
lbl 12
//ld        sm0.0
movw  *vd712,*vd700           // data from VBxx  to txBuffer
incd        vd712             // sourcePointer
incd        vd712
incd        vd700             // destinationPointer
incd        vd700
incw        vw724             // txCount
incw        vw724
decw  vw732             // dataCount (words)
ldw>= vw732,1
jmp   12
ret
```

# B PCS topline midi & PCS plus/win Expander driver

```
// ******* wr-job ******************************************************
sbr 13
ldw=  vw730,0x0020            // wr-job
lbl 13
incd        vd708             // pointer to data
incd        vd708
movw  *vd708,*vd712           // data from rxBuffer to VBxx
incd        vd712
incd        vd712
decw  vw732             // wordCount
ldw>= vw732,1
jmp 13
ret
// *******and-job *****************************************************
sbr 14
ldw=  vw730,0x0040            // AND-job
lbl 14
incd        vd708             // pointer to data
incd        vd708
andw  *vd708,*vd712           // data from rxBuffer to VBxx
incd        vd712
incd        vd712
decw  vw732             // wordCount
ldw>= vw732,1
jmp 14
ret

// *******or-job ******************************************************
sbr 15
ldw=  vw730,0x0080            // OR-job
lbl 15
incd        vd708             // pointer to data
incd        vd708
orw         *vd708,*vd712           // data from rxBuffer to VBxx
incd        vd712
incd        vd712
decw  vw732             // wordCount
ldw>= vw732,1
jmp 15
RET


// *******FILL  RX-BUFFER  ********************************************
int 8                               // receive a byte
movb  smb2,*vd704       // rxChar to rxBuffer
incd        vd704             // pointer +1
incw        vw720             // rxCounter+1
ab>=  vb721,vb651       // until rxCnt >= dataRxCount
=           m6.0
ldb=        vb650,0x02              // STX ?
not
jmp   20
//ldb>=     vb651,15
//jmp 20
ld          sm0.0
jmp   21

lbl 20                              // rcBuffer: error
r           m6.0,1
movb  0xff,vb651
movw  0x0000,vw20
movd  &vb650,vd704            // set rx-pointer
lbl 21
reti                                // end of interrupt
```

# C PCS topline - MPI Direct driver

## C1 General notes

This reference manual exclusively refers to the use of the Profibus-MPI PCS 812 Multibox in connection with the SIEMPIMD PCS maxi driver or SIEMPIMD.DRV or S7MPIMSD.DRV PCSmidi/mini/micro driver, the PCSMPIS.AWL PLC handling software and an S7 of 300/400 programmable controller (Siemens).
The network set-up was tested with a S7 CPU 314. For this configuration, the commissioning is described in the following.
The PCS 812 MPI Multibox software is based on the Siemens SPC2 chip and the MPI documentation. No responsibility can be taken over for errors in this documentation. The programming of the Siemens programmable controller is assumed as known.
The ”SIEMPIMD” driver allows the free allocation of PCS data words on arbitrary addresses of a programmable controller. Select ”MPI Multi-access” in the start selection.
The ”S7MPIMSD” driver permits the free assignment of the PCS data transfer area to optional addresses in one of five programmable controllers. Select ”MPI Multi-PLC” in the start selection.

MPI ( Multi-point interface) is the Siemens programming and communication protocol for all S7 300/400 controllers. It is based on the physical Profibus network with 187,500 bauds and uses the fundamental Profibus routines (layer 2). But it is not part of DIN 19245 (Profibus).

Necessary devices and accessories

The following products by Systeme Lauer are needed for a Profibus MPI set-up

- PCS 812 Multi-box Profibus MPI, version PX812 1003 or higher
- A PCS maxi or midi operating console
- The PCSPRO$^{PLUS}$, PCSPRO$^{WIN}$ or PCSPRO configuration software and a PCS 733 programming cables for the PCS operating console
- This reference manual inclusive drivers diskette

  ... and power supplies for all components

The following Siemens products are needed for a Profibus MPI network set-up

- one S7300 or S7400 programmable controller
- programming software for the programmable controller and the master board
- Profibus network cable and bus interfacing connector

  ... and power supplies for all components

The settings of the components must match so all parts work correctly together.

# C PCS topline - MPI Direct driver

## C2 Configuration

Fundamental data exchange

The communication link uses the following communication partners between PCS and PLC data area:

PCS operating console

↓

PCS 812 Multibox Profibus-MPI

↓

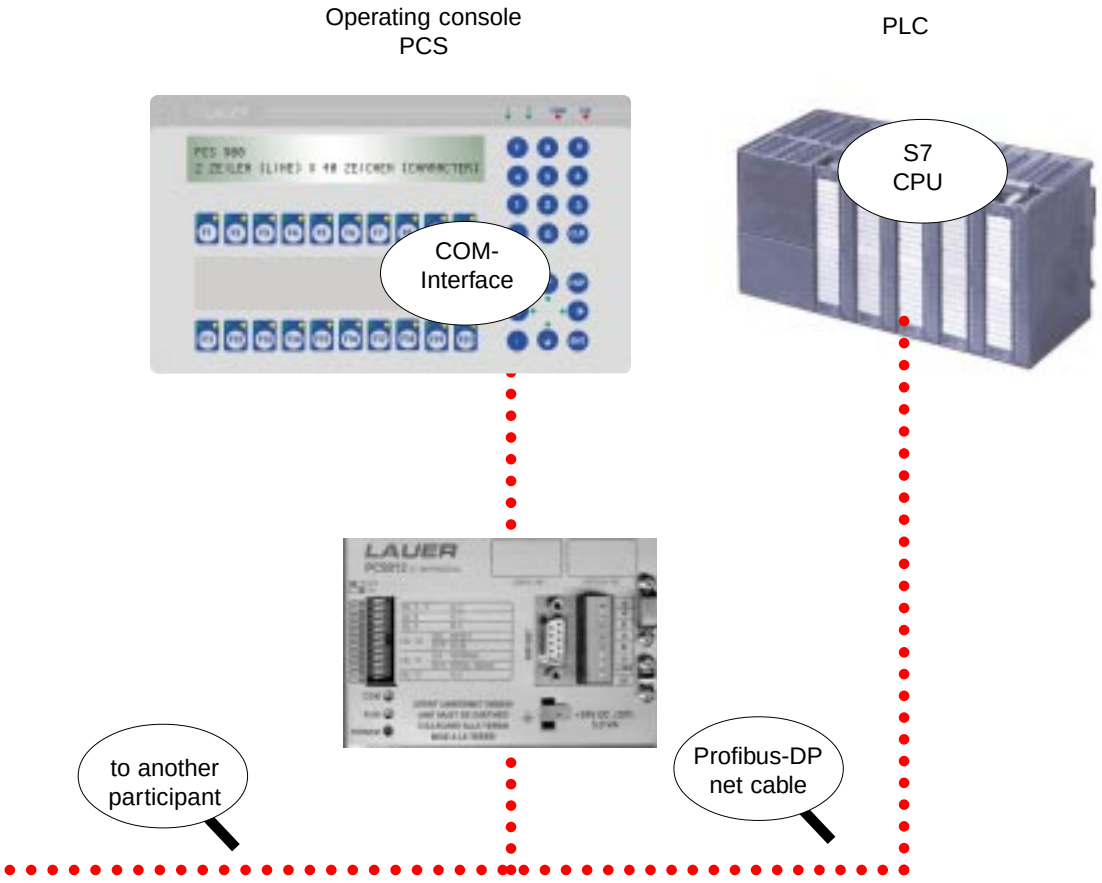
PLC

↓

PCS 812 Multibox Profibus-MPI

↓

PCS operating console



# C PCS topline - MPI Direct driver

The PCS operating console establishes the tasks to be performed (based on the network configuration, the enables, the display contents and the keys) and sends this to the PCS 812 Multibox in one package.

↓

The PCS 812 Multibox creates the access tasks for the CPU

↓

The communication data block with up to 512 words (512 words PCS maxi only) is located In the programmable controller.
Accesses are performed onto this or other variables (e.g. flags)

↓

The PCS 812 Multibox transfers the answer to the PCS

↓

The PCS operating console evaluates the answer and shows the data on the display

## C2.1 Setting up the Multibox

The PCS 812 Multibox is parameterized by the PCS driver. In the PCS, all relevant parameters are set and transferred to the Multibox on powering-up the PCS. Only after this initialization, the Multibox starts accessing the MPI network.
DIL switches 1-9, 12 have no meaning; DIL switch 10 and 11 are used for loading the module firmware. Since the MPI module is already loaded on delivery, all DIL switches have no meaning for you.
The yellow RUN LED lights (should always be on) if supply voltage is applied to the MPI module.
If MPI communication runs correctly, the green COM LED lights too.
This can only be the case if the module was configured by the PCS and a connection to the S7CPU has been established.

Starting with version V1001, you can set the HSA (Highest Station Address) via DIL 8, 9. Select the HSA as large as necessary. Example: the highest participant address is 20 so set the HSA to 31:

| | | | | |
|---|---|---|---|---|
| DIL 8 | OFF | ON | OFF | ON |
| DIL 9 | OFF | OFF | ON | ON |
| HSA | 127 | 63 | 31 | 15 |

Parameterizing the PCS maxi

When configuring the PCS, both the application program and a selected driver are transferred with the data. For adapting the MPI operation, the pre-initializing values of the driver variable can be changed. These are transferred during the communication start of the Multibox.

# C PCS topline - MPI Direct driver

Variables[ COM_TIMEOUT]- PCS time-out time

The time-out time to be set is the time in which the PCS expects an answer to a job package from the Multibox. In the synchronization mode (refer to COM_MODE), it is the time in which the PCS waits for the inverting of word 3. If several participants are on the MPI network, the PCS access becomes slow and the time-out time has to be increased. The time is settable between 2 and 9.9 seconds - default is 4 seconds.

Variables[ COM_PLC_TOUT]- MPI time-out

The time-out time to be set is the time in which the Multibox expectes an answer to a job package from the CPU. If several participants are on the MPI network, the access becomes slowly and the time-out time has to be increased.
The time is settable between 1 and 9.9 seconds - the default setting is 2 seconds.

Variables[ COM_PLC_NUM]- Programmable controller station number

Usually, the programmable controller station number to be set is located on address 2 but can be changed via the PU. The station number is settable between 0 and 127 - default is station 2.

Variables[ COM_PCS_NUM]- Station number PCS

The PCS 812 Multibox station number to be set. The station number is settable between 0 and 127 - default is station 3.

Variables[ COM_MAXDW]- Access to 256 or 512 words

Using this variable you select whether the PCS communication DB should have 256 or 512 words (default value). Ensured that the DB has exactly this minimum length in the PLC. With 512 words, at most 511 words can be occupied.

Variables [ COM_MODE0]. . [COM_MODE3] Access mode

The rotary switch at the rear of the PCS can be set to the positions 0..3.

corresponds to
Position 0=COM_MODE0
Position 1=COM_MODE1
Position 2=COM_MODE2
Position 3=COM_MODE3

Every COM_MODE variable can be assigned one of 3 definitions. The pre-assignment is as follows.

COM_MODE0 = definition 0
COM_MODE1 = definition 1
COM_MODE2 = definition 2
COM_MODE3 = definition 0

# C PCS topline - MPI Direct driver

**The definitions have the following meaning**

Definition 0 = ”NO SYNCRONISATION”

Processing is performed unsynchronously in the CPU, i.e. accesses may occur „now and then“ in the CPU. This can cause undesired effects with accesses across several bytes, e. g. incorrect displays or mutual overwriting of values. Even bit accesses can mutually overwrite each other.

You should have no problems if you define clear accesses to all variable (e. g. variable 27 writes to PCS only, variable 28 writes to PLC only.

**This access mode has advantages**

- It is fast
- No modules are required in the PLC; only the PCS DB must be available

**But there are disadvantages as well**

- Accesses occur unsynchronized
- No time-out monitoring is possible in the PLC and PCS. The PLC cannot determine whether a PCS is connected or not. Conversely, the PCS can not determine whether the CPU is in STOP mode.

Definition 1 = ”PCS LIVE WRITE”

Unsynchronized processing occurs in the CPU, i.e. accesses in the CPU occur ”now and then”.

In addition, a continuous job number is written to word 3. This permits the evaluation of the PCS in the PLC.

**This access mode has advantages**

PCS time-out checking is possible in the PLC

No modules are required in the PLC; only the PCS DB must be available

**But there are also disadvantages**

Accesses occur unsynchronized

Somewhat slower than” NO SYNC”

No time-out monitoring is possible in the PCS. Conversely, the PCS can not determine whether the CPU is in STOP mode.

# C PCS topline - MPI Direct driver

Definition 2 = ”SYNCHRONISATION”

Synchronized processing occurs in the CPU, i.e. accesses occur alternating in the CPU between the PCS and the PLC program. Therefore, you need the ”PCSMPIS.AWL” SYNC modules in the programmable controller. The PCS writes word 3 with low byte = high byte into the PLC. Thus, the PCS concludes its access and the user program may access the PCS DB. After the access, low byte of word 3 is inverted and the PCS accesses again.

**This access mode has advantages**

- Accesses are synchronized
- Time-out monitoring is possible in the PLC and PCS

**But there are also disadvantages**

- It is clearly slower than ”NO SYNC” or ”PCS LIVE WRITE”
- Synchronization modules are necessary in the CPU. These are supplied by Systeme Lauer.

Variables [COM_LIST_MPI]

Cross-reference list

With this table, you assign an address in the PLC to each word addressable in the PCS9092 or PCSPRO$^{PLUS}$. For example, a variable on word 200 can be assigned to flag word 1000 in the programmable controller.

The table is pre-assigned that is starting with word 0 → DB50, DW0 up to word 511 → DB50, DW1022.

Please note that only byte addressing is used in the S7-300/400 and the PLC addresses have therefore to be doubled (thus, they are always even-numbered).

If you create a WORD ARRAY DB50 with 0..511, nothing needs to be altered in COM_LIST_MPI or you do not have to define within the PCS9092.

**Usable areas in the PLC are**

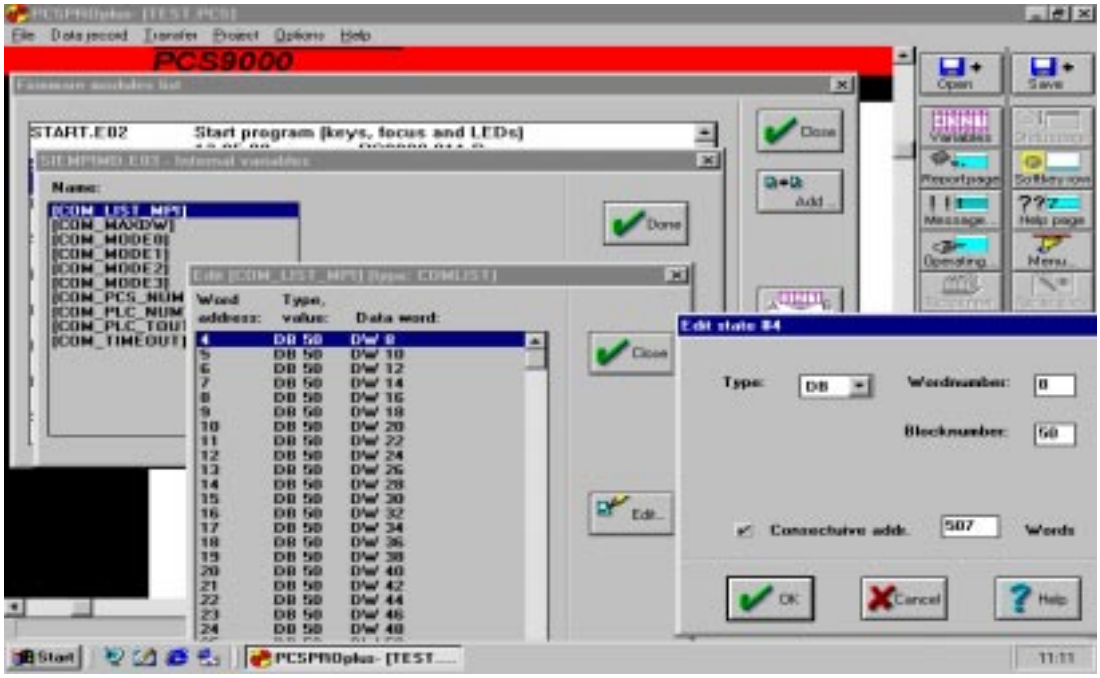
DB words, flag words, input words, output words and timers.

**The formats for PCS 9092 inputs are (examples)**

| | |
|---|---|
| 511 DB50, DW24 | assigns word 511 to DB 50, byte 24 (high byte) and 25 (low byte) |
| 23 TIMERS 25 | assigns word 23 to timer 25. In the configuration software, please create a timer variable at word 23 |
| 77 MW 1000 | assigns word 77 to flag bytes 1000 and 1001 |
| 256 AW 20 | assigns 256 to output bytes 20 and 21 |
| 4 EW 34 | assigns word 4 (F keys) to input bytes 34 and 35 |

In the PCS9092, the COM_LIST_MPI table is to be created in a separate INC file.

# C PCS topline - MPI Direct driver




Address cross-references PCSPRO$^{PLUS}$

Parameterizing PCS micro/mini/midi

When configuring the PCS, both the user program with data and a selected driver will be transferred. To adapt the MPI operation, the pre-initializing values of the driver variable can be changed. AT communication start, these are transferred from the PCS to the Multibox.

Variable AA

PCS time-out time
The time-out time defines the max. permitted time for the answer of a job package from the MPI module. If several participants are on the MPI network, the PCS access becomes slow and the time-out time has to be increased.
The time is settable between 2 and 9.9 seconds. The default is 8 seconds.

Variable AH

PCS station number
The PCS 812 Multibox station number to be set. This is settably between 0 and 127. Default is station number 1.

Variable AO

PLC station number (only SIEMPIMD)
Usually, the programmable controller station number to be set is located at address 2 but can be changed via the PU. The station number can be defined between 0 and 127. Default is station number 2.

Variable BB

MPI time-out
The time-out time to be set is the time in which the PCS expects an answer to a job package from the CPU. If several participants are on the MPI network, the access slows down and the time-out time has to be increased.
The time is settable between 2 and 9.9 seconds. The default is 8 seconds.

# C PCS topline - MPI Direct driver

Variable AC, AD, AE and AF

Access mode

The access mode is set using DIL switches 5 and 6 of the PCS. You can select between the definitions set in PCSPRO or PCSPRO$^{WIN}$ under the Driver Parameters menu item. The default settings are represented in the table below.

| DIL 5 | DIL 6 | Variables | preset mode |
|---|---|---|---|
| OFF | OFF | AC | NO SYNCHRONIZATION |
| ON | OFF | AD | PCS LIVE WRITE |
| OFF | ON | AE | SYNCHRONIZATION |
| ON | ON | AF | NO SYNCHRONIZATION |

In the following table, all definitions are shown that can be assigned to the DIL switches.

**Definitions**

- NO SYNCHRONIZATION
- PCS LIVE WRITE
- SYNCHRONIZATION

Definition

”NO SYNCHRONIZATION”

Processing is performed unsynchronously in the CPU, i.e. accesses may occur „now and then“ in the CPU. This can cause undesired effects with accesses across several bytes, e. g. incorrect displays or mutual overwriting of values. Even bit accesses can mutually overwrite each other.

You should have no problems if you define clear accesses to all variables (e. g. variable 27 writes to PCS only, variable 28 writes to the PLC only.

**This access mode has advantages**

- It is fast
- No modules are required in the PLC; only the PCS DB must be available

**But there are disadvantages as well**

- Accesses occur unsynchronized
- No time-out monitoring is possible in the PLC and PCS. The PLC cannot determine whether a PCS is connected or not. Conversely, the PCS can not determine whether the CPU is in STOP mode.

# C PCS topline - MPI Direct driver

Definition

”PCS LIVE WRITE”
Unsynchronized processing occurs in the CPU, i.e. accesses in the CPU occur ”now and then”.
In addition, a continuous job number is written to word 3. This permits the evaluation of the PCS in the PLC.

**This access mode has advantages**

- PCS time-out checking is possible in the PLC
- No modules are required in the PLC; only the PCS DB must be available

**But there are disadvantages as well**

- Accesses occur unsynchronized
- Somewhat slower than” NO SYNC”
- No time-out monitoring is possible in the PCS. Conversely, the PCS can not determine whether the CPU is in STOP mode.

Definition

”NO SYNCHRONIZATION”
Synchronized processing occurs in the CPU, i.e. accesses occur alternating in the CPU between the PCS and the PLC program. Therefore, you need the ” PCSMPIS.AWL” SYNC modules in the programmable controller. The PCS writes word 3 with low byte = high byte into the PLC. Thus, the PCS concludes its access and the user program may access the PCS DB. After the access, low byte of word 3 is inverted and the PCS accesses again.

**This access mode has advantages**

- Accesses are synchronized
- Time-out monitoring is possible in the PLC and PCS

**But there are disadvantages as well**

- It is clearly slower than ”NO SYNC” or ”PCS LIVE WRITE”
- Synchronization modules are necessary in the CPU. These are supplied by Systeme Lauer.

For the S7MPIMSD(multi-PLC) driver, the additionally 187.5/ 500/1500 Kbauds definition variables are available.
Use 187.5 Kbauds for a pure MPI connection. The 500/1500 Kauds options are available if you want to run a MPI connection in addition to the Profibus-DP.

Attention!
MPI connections above 500/1500 Kbauds are not authorized by Siemens.

# C PCS topline - MPI Direct driver

## C2.2 Address reference list

Variable QVL

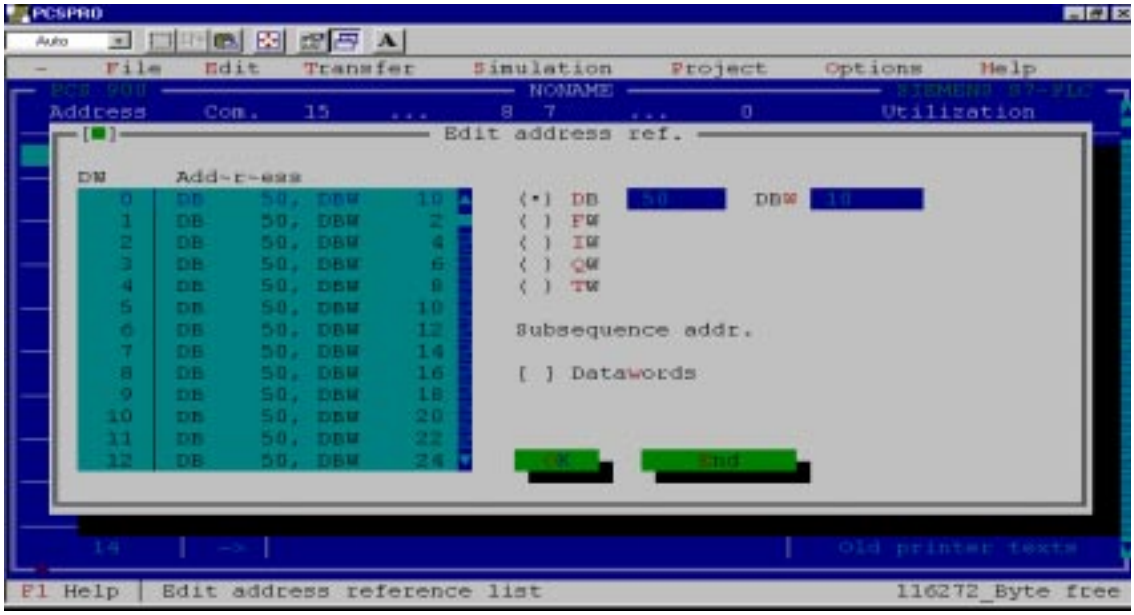
Using this list, you assign an address in the PLC to each word addressable in the PCSPRO (PCSPRO$^{WIN}$).

- For example, a variable using word 200 can be assigned to flag word 1000 in the PLC
- The table is pre-assigned using word 0 ? DB50, DW0 up to word 255 → DB50, DW511
- Please notes that only byte addressing is used in the S7-300/400 and the PLC addresses have therefore to be doubled (thus, they are always even-numbered)
- If you create a WORD ARRAY DB50 with 0 ... 255 in the PLC, nothing needs to be altered in the cross-reference list or there is no need to create it in the PCS configuration program

**Usable areas in the PLC are**
DB words, flag words, input words, output words and timers.

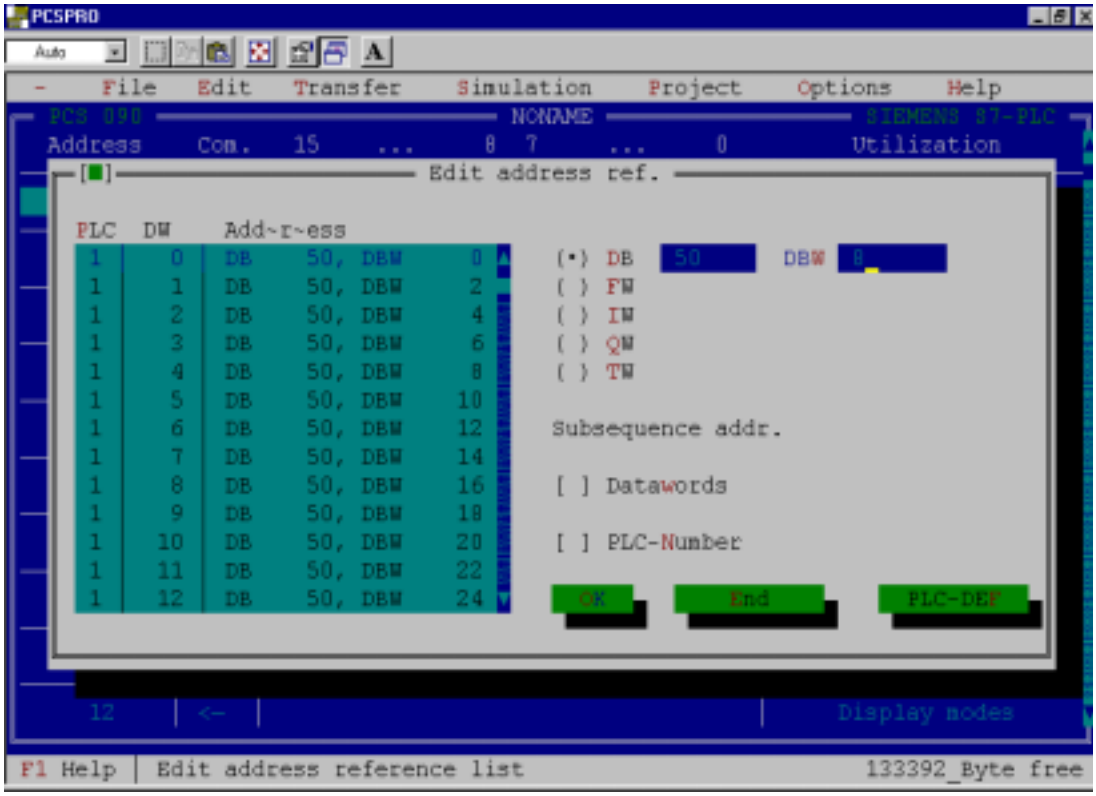
PCSPRO and PCSPRO$^{WIN}$

These allocations can be changed under the Address References menu item or in the PLC Transfer Area menu item by double-clicking onto the respective item.

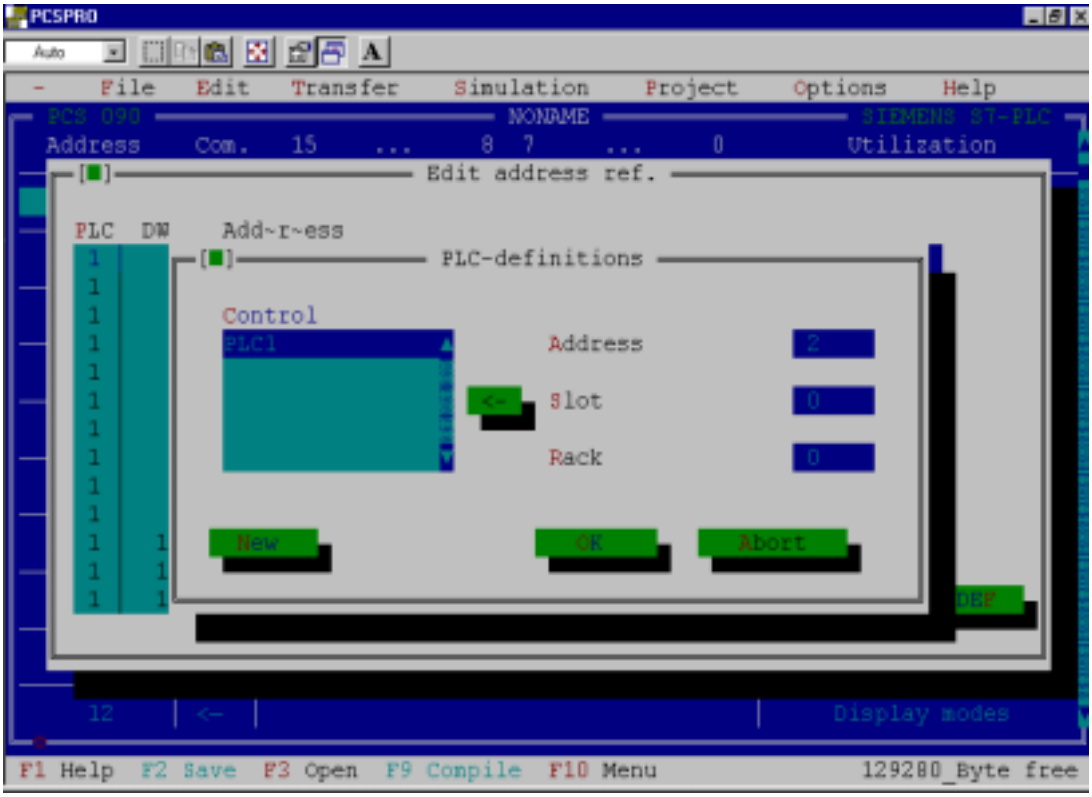



SIEMPIMD PCSPRO driver address references for 1 PLC

# C PCS topline - MPI Direct driver

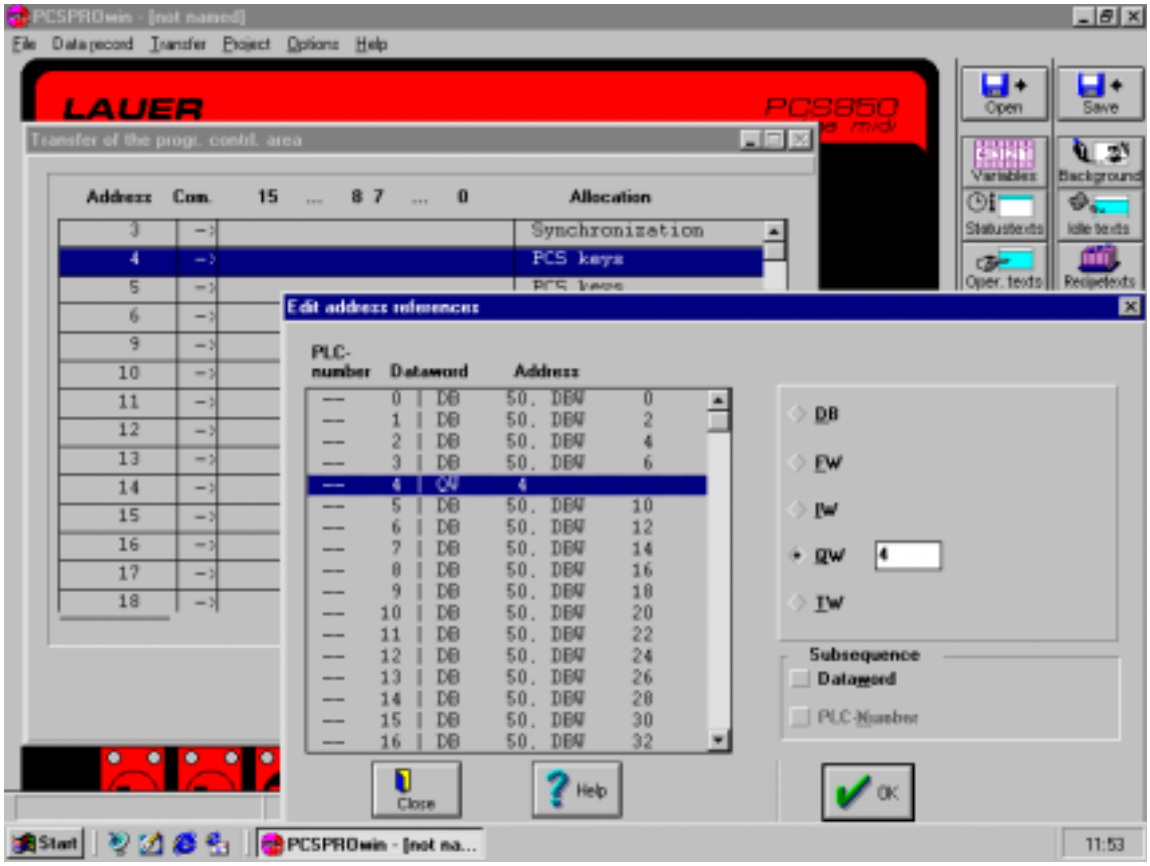



S7MPIMSD PCSPRO driver address references

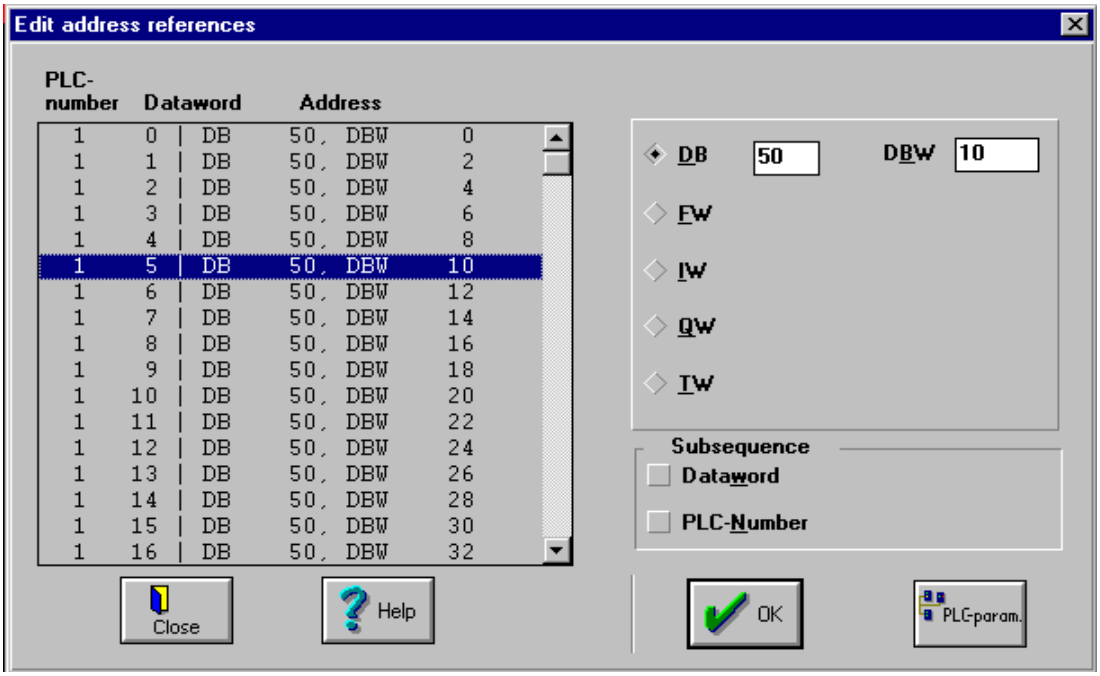



Definition of the station number (a PLC, here PLC 1) for the S7MPIMSD PCSPRO driver for 1-5 PLCs.

# C PCS topline - MPI Direct driver

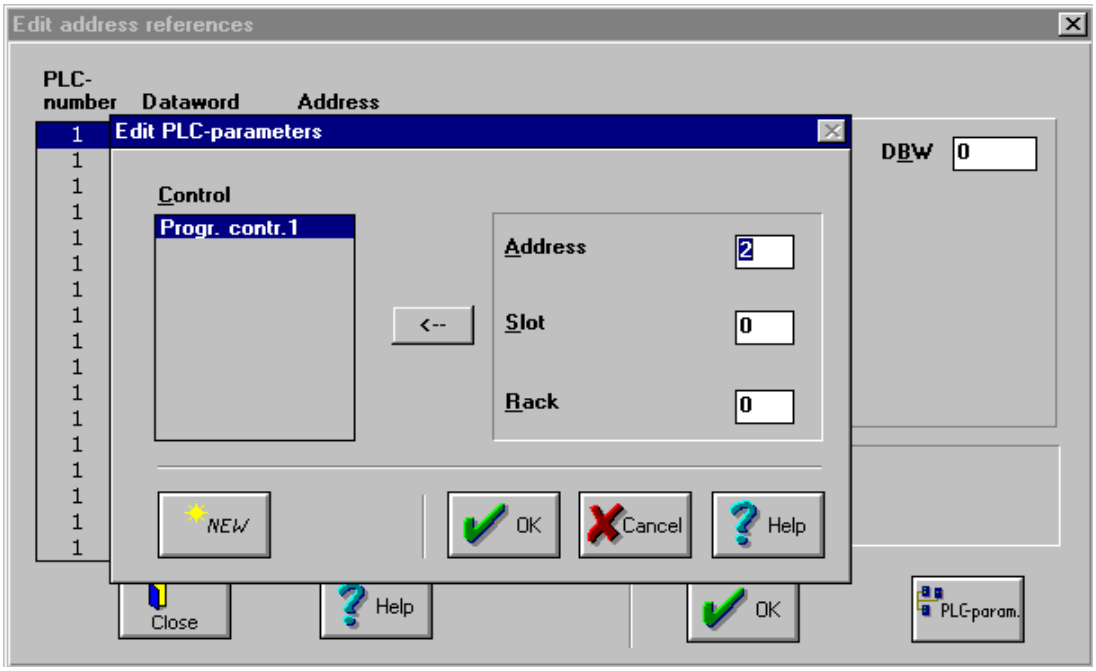



SIEMPIMD PCSPRO$^{WIN}$ driver address references for 1 PLC




S7MPIMSD PCSPRO$^{WIN}$ driver address references for 1 - 5 PLCs

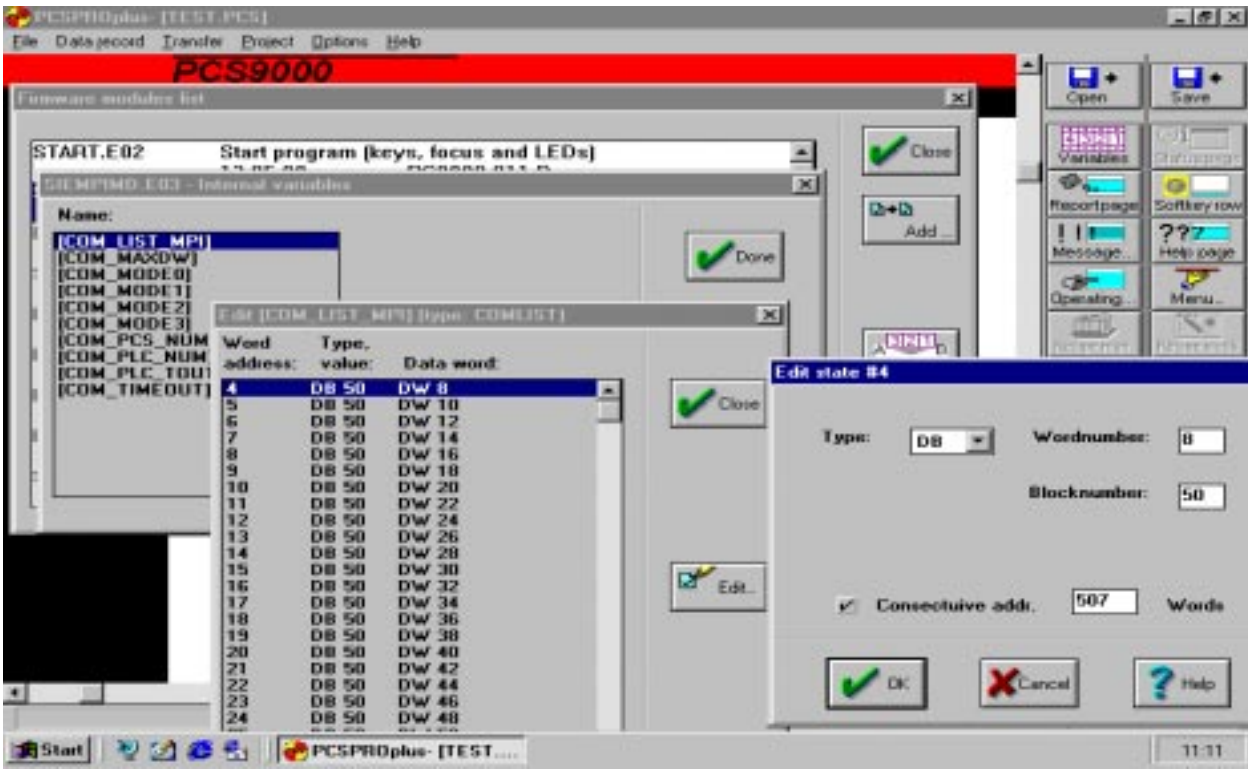
# C PCS topline - MPI Direct driver




Definition of the station number for one PLC, here PLC 1, for the S7MPIMSD PCSPRO$^{WIN}$ driver for 1 - 5 PLCs.

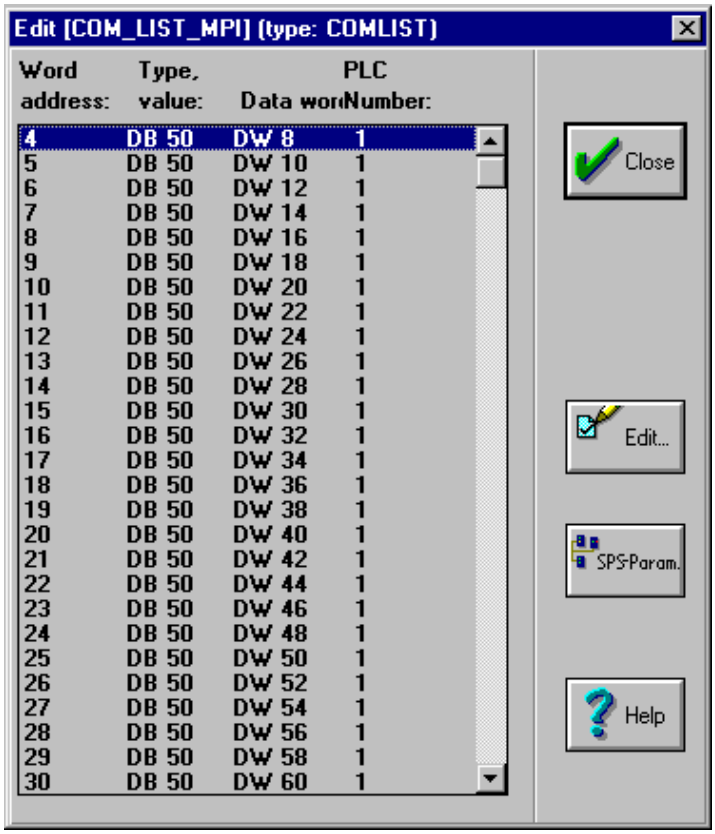
PCSPRO$^{PLUS}$ and PCS9092

The address reference list is edited using the internal [COM_LIST_MPI] variables of the SIEMPIMD driver.

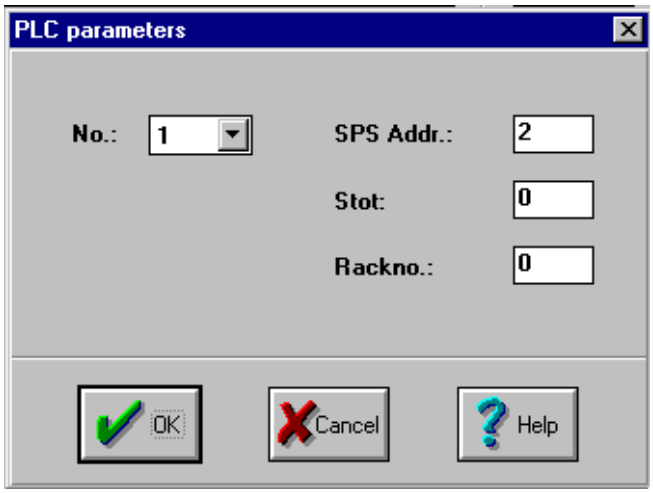



PCSPRO$^{PLUS}$ address references

# C PCS topline - MPI Direct driver




PCSPRO$^{PLUS}$ address references




Definition of the station number for 1-5 PLCs, here PLC 1

# C PCS topline - MPI Direct driver

Example PCS9092

DESCRIPTIONEND

PCS9000_PARAMETER

```
BEGIN

LANGUAGEMAX, (2)
CHARSET (1), 437, 850
MSG_WINDOW, FONT (1), (20), 4
MSG_DW_RANGE, 25, 10
STATUS_WINDOW, (160)
SKEYHIGH, (50)
DATE_SIZE, EU

 // internal variables definitions

INTVAR, [COM_LIST_MPI]
3, DB 50, DW 6
4, TW 8
5, DB 50, DW 10
6, DB 50, DW 12
7, DB 50, DW 14
8, DB 50, DW 16
9, DB 50, DW 18
10, DB 50, DW 20
11, DB 50, DW 22
12, DB 50, DW 24
13, DB 50, DW 26
14, DB 50, DW 28
15, DB 50, DW 30
16, DB 50, DW 32
17, DB 50, DW 34
18, DB 50, DW 36
19, DB 50, DW 38
20, DB 50, DW 40
21, DB 50, DW 42
22, DB 50, DW 44
23, DB 50, DW 46
24, DB 50, DW 48
25, DB 50, DW 50
```

# C PCS topline - MPI Direct driver

## C2.3 Optimal configuration

The communication speed mainly depends on the following factors.

Required number of jobs

in the PCS
The jobs of the PCS operating console depend on the display contents and the enabled transmissions in the command words.

Assignment

With an increased number of changes of the default values of the linearly addressed list [ COM_LIST_MPI or QVL], fewer jobs can be combined. In the worst-case case, communications becomes 4 times slower so that a jog operation is not sensible any more. Even splitting the accesses onto several PLCs decreases the communication speed!

Access modes

The [COM_MODE] variable (PCS maxi) or variables AC to AF (PCS midi). The fastest mode is ”NO SYNCRONISATION”. You can obtain a key LED time of approx. 0.5 seconds (if COM_LIST_MPI or QVL were not fragmented).
”PCS LIVE WRITE” is approx. 25% slower than ”NO SYNC” since word 3 is written in addition.
”SYCRONISATION” can become very slow, for example, in case of extensive PLC cycles as the PCS waits on the enable of the PLC program. At the minimum, „SYNCRONISATION” is approx. 50% slower than ”NO SYNC”.

**Configuration examples of command words**

PCS 950 — The command words in the PLC are assigned as follows

W36=KH0F60
W37=KH0001

The transmission of time and date is blocked and 1 message block is released for transfer. Now, if you take care that there are just a few variables in the display then you have optimal communications.

PCS 095 — The command word in the PLC is assigned as follows

W13=KH0FC1

Thus, the messages M0..31 are released for transmission. Now, if you take care that there are just a few variables in the display then you have optimal communications.

PCS 9000 — PLC command words

W13=KH 000C (CLK_DBit on pos.2) (CLK_CBit on pos.3)
W16=KH0
W17=KHFF00

Therefore, time and date are disabled. If possible, do not include external variables in the status window.

# C PCS topline - MPI Direct driver

## C2.4 Transfer of the data record into the PCS

1. Apply operating voltage (19...33 V) to the PCS. The ERR LED lights now.
2. Connect the programming interface of the PC to the PCS operating console via the PCS 733 programming cable.
3. Run the PCS 9092, PCSPRO$^{WIN}$ or PCSPRO configuration software.
4. Select the suitable MPI driver.
5. In your PCS project, set the ([COM..] for PCSmaxi) driver variable to the desired values.
6. Compile and transfer the firmware or data modules into the PCS.
7. Set the rotary switch to position 0, 1 or 2 (or the corresponding DIL switches of the PCSmidi) and start the PCS.
8. Set up the MPI network. The COM LED on the PCS is deactivated as soon as the communication connection has been established; the green COM LED on the PCS 812 module must light now. Otherwise, you will receive a detailed error message on the PCS.

## C2.5 Set-up and the first powering-up

Set up the assembly without applying power after you have configured all parts.

**The following points should be taken into consideration**

Profibus-MPI network

Interfacing is possible only on the MPI network that includes the target CPU.

- Use only suited cables for wiring. The distance between participants may not be longer than 50 m.
- The last participant on the MPI network must have a terminator installed.
- Use Siemens ”SINEC L” bus interfacing connectors for that.
- On the PCS 812 Multibox, you can also use the supplied Lauer terminals instead of the Siemens connectors.
- If the Lauer terminals are used then the red cable is connected to ”A” and the green cable to ”B”. The cable screening is grounded at the cable clamp.
- A simultaneous operation of the programming unit (PU) and PCS is possible.
- The connection of several PCS to one CPU is possible. With an S7-300 CPU, a maximum of 3 PCS and one PU can be connected at the same time.
- Then, they can access the various DBs to a full extent.
- Or in the ”NO SYNC” mode, you can access the same DB using several operating consoles operating in parallel.
- Set the HSA via DIL switches 8, 9. The HSA should be selected as small as possible. E. g. is your highest participant address (station number) 20 then select HSA=31.

# C PCS topline - MPI Direct driver

**Proceed with the powering-up as follows**

- Switch on the PLC and the PCS 812 Multibox.
- Connect the PCS (COM interface) to the PCS 812 Multibox.
- If you work in the Sync mode, you should set the restart input at the PLC to „ON" or switch the PLC from STOP mode to RUN mode.
- After at the latest 3 seconds, the COM LED of the PCS should be deactivated; however, the COM LED of the PCS 812 should light. If this is not the case continue reading in the Troubleshooting section.

The pin layout of the 9-pin MPI female connector of the PCS 812 corresponds to the Profibus-DP.

| Pin No. | Signal name | Designation |
|---|---|---|
| 1 | - | |
| 2 | - | |
| 3 | RS-485 | Data line B |
| 4 | RTS | Request to send |
| 5 | 5V ground | external |
| 6 | + 5V | external |
| 7 | - | |
| 8 | RS-485 | Data line A |

**Mechanical mounting of the PCS 812 MPI Multibox**
The PCS 812 Multibox can be mounted on a top-hat rail. The PCS operating console and the PCS 812 Multibox have to be grounded.

## C2.9 Trouble-shooting

PCS COM LED remains on

No COM error message is displayed

- There is no connection to the PCS812 Multibox. Check the connection.
- Does the PCS812 RUN LED light
- Has been the correct driver being loaded into the PCS (SIEMPIMD or SIEMPIDD are correct)
- Reset the PCS812 module
- For test purposes, remove the MPI connection at the PCS812 module

PCS COM LED flashes

A COM error message is displayed

- There is a connection to the PCS812 Multibox; however, an error occurred in the link to the target programmable controller

MPI TIME-OUT ERROR

The programmable controller number was not found

- Check the PLC and the PCS number in the driver variables

# C PCS topline - MPI Direct driver

| | |
|---|---|
| MPI ACCESS ERROR | An access in the PLC failed<br>• Check all words from the COM_LIST_MPI whether these are available or whether the DB was created sufficiently large. |
| SYNCRONIZATION ERROR | Sync word 3 is not processed in the PLC. This error message is only generated in the ”SYNCRONIZATION” access mode.<br>• Did you load the ”PCSMPIS” SYNC software into the PLC<br>• Is the PLC in RUN mode<br>• Is the restart input in the PLC set to ”ON” level<br>• Is the correct word processed in the PLC (check the cross-reference list) |
| TIME-OUT COMMUNICATION | The PCS has no connection to the PCS 812 Multibox anymore<br>• A problem in the MPI connection is probable<br>• Test the MPI connection<br>• Is the PLC still running<br>• Has a further participant blocked the MPI line<br>• Reset the PCS 812 module |
| SIEMPIMD driver | Error messages<br>In order to facilitate the handling of the MPI interfacing, several error messages of the driver are issued. These always appear in the ”COMMUNICATION ERROR” error window together with a supplementary text. |
| TIME-OUT COMMUNICATION | The PCS has no connection to the PCS 812 Multibox (anymore) |
| MPI TIME-OUT ERROR | The connection with the specified PLC address was lost or the PLC address is not available |
| MPI ACCESS ERROR | The access to a PLC word failed<br>• Check all words from the COM_LIST_MPI whether these are available or whether the DB was created sufficiently large |
| MPI MODULE ERROR | There was an internal module error<br>• Inform the Systeme Lauer support |
| NO MPI MODULE | The connected module is not an MPI module<br>• Use the PCS812 |

# C PCS topline - MPI Direct driver

SYNCRONIZATION ERROR

Sync word 3 is not processed in the PLC. This error message is only generated in the ”SYNCRONIZATION” access mode.

- Did you load the ”PCSMPIS” SYNC software into the PLC
- Is the PLC in RUN mode
- Is the restart input in the PLC on ”ON” level
- Is the correct word processed in the PLC (check the cross-reference list)

System requirements MPI network for S7-300 and S7-400

# C PCS topline - MPI Direct driver

## C3 PCS 812 Specifications

| | |
|---|---|
| Mounting dimensions | Height: 50mm,<br>Width: 80mm,<br>Length: 120mm (without cable) |
| Supply voltage | 24 volts± 10% |
| Current consumption (@ 20° C) | max. 200 mA |
| Power consumption | 5 VA max. |
| Working temperature range | 0. . +50 °C |
| Storage temperature range | -20. . +80 °C |
| Interfaces | • 25-pin sub D female connector with RS-232 interface to the PCS operating console<br>• 9-pin sub D male connector with RS-485 Profibus interface<br>• 8-pin terminal block with 24 volts supply voltage and RS-485 Interface |
| Indicators | • 1 yellow LED for supply voltage (lighting = voltage ON)<br>• 1 yellow LED for LOAD/RUN (state lighting = RUN)<br>• 1 green LED for the communication status (lighting = active MPI comm.) |
| DIL switches | • DIL switches 1..7 and 12 are without meaning<br>• DIL 10 as Reset switch (ON = Reset)<br>• DIL 11 for switching between LOAD (=OFF) and RUN (=ON) |

Note!
There are 2 firmware versions for the PCS 812 MPI Multibox:

- Firmware version 000.3 for connecting to one PLC (old)
- Firmware version 100.2 for connecting several PLCs (current)

Version 100.2 is valid for S7MPIMD and S7MPIMSD.
Starting with version 1001, the HSA (highest station address) can be set via DIL switches 8 and 9.

| | | | | |
|---|---|---|---|---|
| DIL 8 | OFF | ON | OFF | ON |
| DIL 9 | OFF | OFF | ON | ON |
| HSA 127 | 63 | 31 | 15 | |

# C PCS topline - MPI Direct driver

## C3.1 Functions of the PCS 812

PCS 812 Multibox Profibus MPI

The PCS 812 Multibox Profibus MPI software is based on the documentation of the Siemens company. Serial communication and logical evaluation were added. Furthermore, the firmware is completely loadable.

Loading state

If DIL 11=OFF (yellow RUN LED OFF) then the PCS 812 Multibox is in the LOADING state, i.e. the EPROM is active and the EEPROM is externally addressable. Using a PC loader, a firmware can now be serially loaded into the device. Normally, this is not necessary as the device is delivered with firmware.

RUN state

If DIL 11=OFF (yellow RUN LED ON) then the PCS 812 Multibox is in the RUN state and executes the EEPROM program (the EPROM is switched off). In order to guarantee a defined start of the software, a reset has to be triggered before the switching over via DIL 10=ON that is reset after the switching over via DIL 10=OFF.
The logical communication between the PCS 812 Multibox and the PLC is started only after mounting the serial interface onto the PCS 812 Multibox.

Visual control

**"Power On" LED, yellow.**

- This LED signals the functioning of the switching power supply

**" RUN" LED, yellow.**

- This LED is OFF in the LOADING state; it is ON in the RUN state

**"Communication" LED, green.**

- This LED is OFF without communications. The LED lights if a data exchange takes place.

This can only be the case if the PCS has configured the module and the MPI connection is correctly established.

# C PCS topline - MPI Direct driver

## C4 Operating software

Structure of the handling software

- For the operation of the PLC with the PCS in **synchronization mode** the „PCSMPIS.AWL" handling software must be loaded into the PLC
- Load PCSMPIS.AWL as an SO object (source) and compile it
- If you use an S7-300, the compiled OB101 should be erased since it can only be used by an S7-400
- In FC100 and FC101, adapted the assignment of the PCS DB to your project
- In FC102, access only the PCS DB
- If you want to increase the time-out time of the PLC monitoring then you do this by passing the value in the TIMZ parameter when calling FC103 (OB1)

Program structure

OB1 ——— FC103 ————— FC101
——- FC102
OB100 ——— FC100

DB50

Attention!
Refer to section 3.3 PCS 812 Specifications on page 3-25.

Programs

**OB1** OB cycle. There, FC103 is called with parameters.

**OB100** Start OB. There, FC100 is called.
(Attention: In case of S7-400, the start OB is OB101)

**FC103** Processes the access to the PCS DB. There, FC101 and FC102 are called.

**FC100** PCS initalization FC. There, the pre-assignments of the PCS DB are made at the PLC start.

**FC101** PCS communication loss FC. There, the measures are defined for a communication loss. This FC is called only once per loss.

**FC102** The PLC program may access the PCS DB here

**DB50** PCS communication DB. Here, the PLC and PCS communicate, 512 words in length. For the PCSmidi, mini, micro series only 256 words in length.

# C PCS topline - MPI Direct driver

Call parameter of the FC103

```
CALL FC  103 (                // call PCS_SYNC
      UBDB  := DB    50,      // USER DB
//PCS DB.
      TIMT  := T      5,      // TIMEOUT TIMER
//PCS Sync Timeout Timer
      TIMZ  := S5T#2S,       // TIMEOUT VALUE
//Timeoutwert 2 Sekunden. Bei vielen MPI Teilnehmern vergr÷ern.
      RSTRT      := M     10.0,     // RESTART FLAG
//ist dieses Bit =1, so l£uft Kommunikation nach Fehler wieder an
      COFF  := FC   101,      // COMMUNICATION ERROR FC
//Kommunikationsfehler FC
      USERPRG     := FC   102,      // USER DB ACCESS DB
//FC, in dem Sie auf den PCS DB zugreifen
      TIMO  := M     10.1)    // COMMUNICATION ERROR FLAG
//Fehlerausgang, 1= Kommunikationsfehler
```

PCSMPIS.AWL listing

**Pre-conditions**

- Words 0. . 3 may not be used in the FC102
- In COM_LIST_MPI, word 3 has to be assigned to UBDB, address 6

```
FUNCTION FC 100: VOID
TITLE =INIT
VERSION : 0.1


VAR_INPUT
  UBDB : BLOCK_DB ;     //PCS-User-DB
END_VAR
BEGIN
NETWORK
TITLE =
//;    Please choose your PCS and add the commands to your network
//;                       // Example PCS900/PCS 920/PCS950
//;                       // Typical presets
//;L W#16#1F00;  //(3F00 PCS950/PCS 920)
//;T DBW 72;
//;L W#16#00FF;
//;T DBW 74;
//;L W#16#0080;
//:T DBW 76;
//;                       // Example PCS009/PCS090/PCS095
//;                       // Typical presets
//;     L W#16#0FC8;
//;     T DBW 26;
//;     L W#16#0080;
//;     T DBW 28;
//;                       // Example PCS9000
//                        // Typical presets
//;L W#16#0000;
//;T DBW 26;
//;T DBW 28;
//;T DBW 32;
//;T DBW 36;
//;L W#16#FF00;
//;T DBW 34;
//
// Clear all keys
// For all PCS-Types
      AUF   #UBDB; // open User-DB
      L     W#16#FF;
      T     DBW    4;
      T     DBW    6;
      L     W#16#0;
```

# C PCS topline - MPI Direct driver

```
      T     DBW    8;
      T     DBW   10;
      T     DBW   12;
      T     DBW   14;
      T     DBW   46;
// Add your default values
END_FUNCTION


FUNCTION FC 101: VOID
TITLE =COFF
NAME : COFF
VERSION : 0.0

BEGIN
NETWORK
TITLE =
//;                     // Presets emergancy case
//;// Clear all keys
      L     W#16#0; // Clear all keys
      T     DBW    8;
      T     DBW   10;
      T     DBW   12;
      T     DBW   14;
      T     DBW   46;
// Add other clearings
END_FUNCTION

FUNCTION FC 102: VOID
TITLE =
AUTHOR : Lauer
NAME : Userprg
VERSION : 0.1

BEGIN
NETWORK
TITLE =

                       [ add your PCS program accesses here!]

      L     DBW    8;   // Copy key
      T     DBW   38;   // to LED for PCS 9000

                              END_FUNCTION

                              FUNCTION FC 103: VOID
                              TITLE =PCS_SYNC
                              AUTHOR  : Lauer
                              NAME    : Sync
                              VERSION : 0.1


                              VAR_INPUT
                                UBDB  : BLOCK_DB ; //USER DB
                                TIMT  : TIMER ;    //TIMEOUT TIMER
                                TIMZ  : S5TIME ;   //TIMEOUT VALUE
                                RSTRT : BOOL ;     //RESTART FLAG
                                COFF  : BLOCK_FC ; //ERROR FC
                                USERPRG : BLOCK_FC ;     //USER DB ACCESS FC
                              END_VAR
                              VAR_OUTPUT
                                TIMO : BOOL ;      //TIMEOUT FLAG
                              END_VAR
                              BEGIN
                              NETWORK
                              TITLE =
```

# C PCS topline - MPI Direct driver

```
      AUF   #UBDB;  //open User DB
      U     #TIMT;
      =     #TIMO;  // Timeout error ?
      U     #TIMO;
      SPBN  KTIM;         // NO -> KTIM
      U     DBX    0.2;   // COFF already called ?
      SPB   KCOF;
      UC    #COFF;  // NO -> call COFF
      UN    DBX    0.2;
      =     DBX    0.2;
KCOF: U     DBX    0.0;   // Restart was set in past ?
      SPB   KTIM;         // Yes -> Test PCS access
      UN    #RSTRT; // Restart now set ?
      U     #TIMT;  // Timeout ?
      BEB   ;       // No RSTRT and Timeout -> End
      U     #RSTRT; // Store RESTART
      =     DBX    0.0;
      L     W#16#FF;// Preset sync words
      T     DBW    4;
      T     DBW    6;
KTIM: L     DBW    4;     // New sync word received ?
      L     DBW    6;
      ==I   ;
      BEB   ;
      UC    #USERPRG;     // Yes -> Access to User-DB
      L     DBB    6;
      INVI  ;
      T     DBB    7;     // Save inverted syncbyte to byte 7
      L     DBW    6;     //
      T     DBW    4;
      U     DBX    0.2;
      R     DBX    0.2;
      UN    DBX    0.2;
      R     DBX    0.0;
      R     DBX    0.1;
      U     DBX    0.1;   // restart timeout timer
      FR    #TIMT;
      U     DBX    0.1;
      L     #TIMZ;
// END FC 103
      SE    #TIMT;
      UN    DBX    0.1;
      S     DBX    0.1;
      U     DBX    0.1;
      L     #TIMZ;
      SE    #TIMT;
END_FUNCTION
ORGANIZATION_BLOCK OB 1
TITLE =
VERSION : 0.1

VAR_TEMP
  OB1_EV_CLASS : BYTE ;         //Bits 0-3 = 1 (Coming event),
Bits 4-7 = 1 (Event class 1)
  OB1_SCAN_1 : BYTE ;           //1 (Cold restart scan 1 of OB
1), 3 (Scan 2-n of OB 1)
  OB1_PRIORITY : BYTE ;         //1 (Priority of 1 is lowest)
  OB1_OB_NUMBR : BYTE ;         //1 (Organization block 1, OB1)
  OB1_RESERVED_1 : BYTE ; //Reserved for system
  OB1_RESERVED_2 : BYTE ; //Reserved for system
  OB1_PREV_CYCLE : INT ;  //Cycle time of previous OB1 scan
(milliseconds)
  OB1_MIN_CYCLE : INT ;         //Minimum cycle time of OB1
(milliseconds)
  OB1_MAX_CYCLE : INT ;         //Maximum cycle time of OB1
(milliseconds)
```

# C PCS topline - MPI Direct driver

```
  OB1_DATE_TIME : DATE_AND_TIME ;     //Date and time OB1 started
END_VAR
BEGIN
NETWORK
TITLE =

//***** PCS SYNCRONISATION FC CALL ***************

      CALL FC   103 (//User Access Test FC
   UBDB   = DB 50,   //User DB
   TIMT  := T  5,    //Timeout Timer
   TIMZ  := S5T#2S,  //Timeout Value
   RSTRT := E  4.0,  //Restart Flag
   COFF  := FC 101,  //Communication Timeout FC
   USERPRG     := FC 102,  //User DB Access FC
   TIMO  := A  0.0); //Communication Timeout Flag

END_ORGANIZATION_BLOCK

ORGANIZATION_BLOCK OB 100
TITLE =
VERSION : 0.1


VAR_TEMP
  OB100_EV_CLASS : BYTE ; //16#13, Event class 1, Entering event
state, Event logged in diagnostic buffer
  OB100_STRTUP : BYTE ;         //16#81/82/83/84 Method of startup
  OB100_PRIORITY : BYTE ; //27 (Priority of 1 is lowest)
  OB100_OB_NUMBR : BYTE ; //100 (Organization block 100, OB100)
  OB100_RESERVED_1 : BYTE ;     //Reserved for system
  OB100_RESERVED_2 : BYTE ;     //Reserved for system
  OB100_STOP : WORD ;           //Event that caused CPU to stop
(16#4xxx)
  OB100_STRT_INFO : DWORD ;     //Information on how system
started
  OB100_DATE_TIME : DATE_AND_TIME ;   //Date and time OB100
started
END_VAR
BEGIN
NETWORK
BEGIN
NETWORK
TITLE =

      CALL FC   100 (// call PCS Initialisation
           UBDB                    := DB    50);

END_ORGANIZATION_BLOCK

ORGANIZATION_BLOCK OB 101     [only for S7-400!]
TITLE =
VERSION : 0.1


VAR_TEMP
  OB101_EV_CLASS : BYTE ; //16#13, Event class 1, Entering event
state, Event logged in diagnostic buffer
  OB101_STRTUP : BYTE ;   //16#81/82/83/84 Method of startup
  OB101_PRIORITY : BYTE ; //27 (Priority of 1 is lowest)
  OB101_OB_NUMBR : BYTE ; //101 (Organization block 101, OB101)
  OB101_RESERVED_1 : BYTE ;     //Reserved for system
  OB101_RESERVED_2 : BYTE ;     //Reserved for system
  OB101_STOP : WORD ;     //Event that caused CPU to stop (16#4xxx)
  OB101_STRT_INFO : DWORD ;     //Information on how system
started
  OB101_DATE_TIME : DATE_AND_TIME ;   //Date and time OB101
```

# C PCS topline - MPI Direct driver

```
started
END_VAR
BEGIN
NETWORK
TITLE =

      CALL FC   100 (// call PCS Initialisation
           UBDB                    := DB    50);

END_ORGANIZATION_BLOCK

DATA_BLOCK DB 50
TITLE =
AUTHOR : Lauer
NAME : PCS_DB
VERSION : 0.1


  STRUCT
   dw : ARRAY  [0 .. 511 ] OF //PCS Communication DB
   WORD ;
  END_STRUCT ;
BEGIN
   dw[0] := W#16#0;
   dw[1] := W#16#0;
   dw[2] := W#16#0;
   dw[3] := W#16#0;
   dw[4] := W#16#0;
  .....
   dw[509] := W#16#0;
   dw[510] := W#16#0;
   dw[511] := W#16#0;
END_DATA_BLOCK
```

# Index

# Index



## Q



## R



## S



## T



## V